Biblioteczka Komputer Świat




**DVD I KŚ+**
- NAJLEPSZE NARZĘDZIA DO ANIMACJI
- PLIKI SZKOLENIOWE

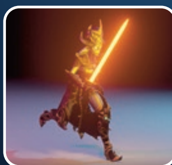
ISO płyty do pobrania z ksplus.pl

# blender

## JAK TWORZYĆ ANIMACJE

### KURS OD PODSTAW






- modelowanie i teksturowanie postaci
- obsługa linii czasu i klatek kluczowych
- łączenie ruchu w zaawansowane sekwencje
- wizualizacja dźwięku
- przykłady z plikami szkoleniowymi




# Z TĄ KSIĄŻKĄ E-WYDANIE GRATIS

**Poniżej znajduje się płyta z kodem bonusowym dającym dostęp do e-wydania tej książki w serwisie KŚ+ (ksplus.pl) oraz pliku ISO z cyfrową wersją płyty do pobrania.**

## NA PŁYCIE DVD

**Płyta dołączona do tej książki zawiera Blender oraz inne najlepsze narzędzia do tworzenia animacji, a także pliki projektów omówionych krok po kroku we wskazówkach.**

---

**Jeżeli brakuje płyty, poinformuj sprzedawcę lub BOK: pomoc@komputerswiat.pl**

Biblioteczka Komputer Świat

blender®

PŁYTA JEST DODATKIEM DO KSIĄŻKI
KOMPUTER ŚWIAT BIBLIOTECZKA
5|2022

i najlepsze narzędzia do animacji

NA PŁYCIE ZNAJDZIESZ KOD DO E-WYDANIA I CYFROWEJ WERSJI PŁYTY W KSPLUS.PL

**Kod bonusowy należy zarejestrować w KŚ+ (ksplus.pl)**

Biblioteczka Komputer Świat

MIKOŁAJ BARTOSZEK

# blender

## JAK TWORZYĆ ANIMACJE

## KURS OD PODSTAW

ringier
axel springer

**AUTOR:** Mikołaj Bartoszek

**REDAKTORZY PROWADZĄCY:** Rafał Kamiński, Agnieszka Al-Jawahiri

**PRZYGOTOWANIE PŁYTY:** Mariusz Michalski

**PROJEKT OKŁADKI:** Robert Dobrzyński

**SKŁAD I ŁAMANIE:** Mariusz Rybak

**KOREKTA:** Jolanta Rososińska

**WYDAWCA:**

RINGIER AXEL SPRINGER POLSKA Sp. z o.o.
02-672 Warszawa, ul. Domaniewska 49
tel. 12 2600200 (BOK)
www.ringieraxelspringer.pl

ISBN 978-83-8250-199-5



Warszawa 2022

**BUSINESS PROJECT MANAGER:** Paweł Bulwan

**DRUK I OPRAWA:**
Drukarnia im. Adama Półtawskiego, Kielce

**EGZEMPLARZE ARCHIWALNE:**
literia.pl, prenumerata.axel@qg.com

**E-WYDANIA:**
ksplus.pl

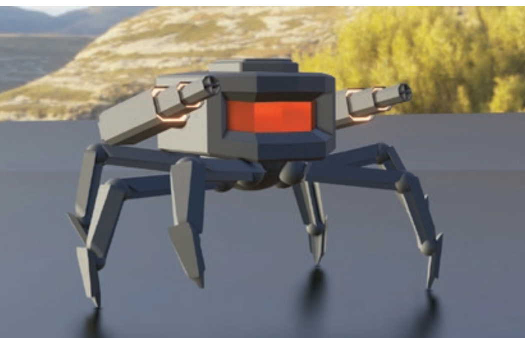
**KONTAKT:**
redakcja@komputerswiat.pl

**INTERNET:**
komputerswiat.pl, ksplus.pl

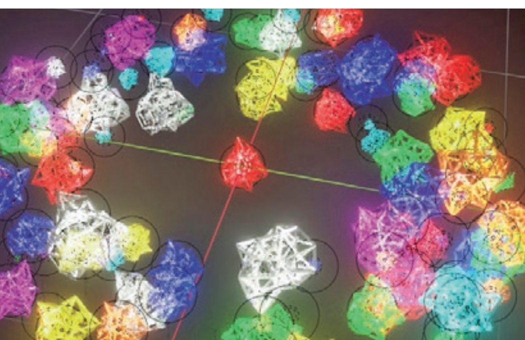
Płyta DVD jest dodatkiem do książki

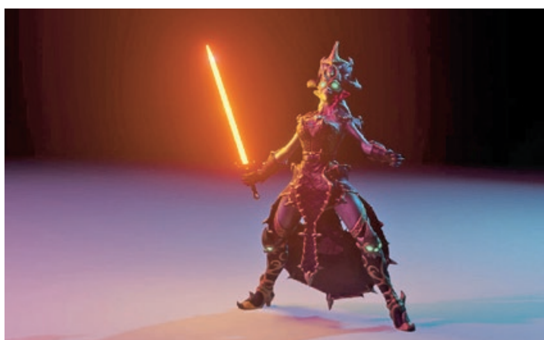
ringier
axel springer

# Spis treści









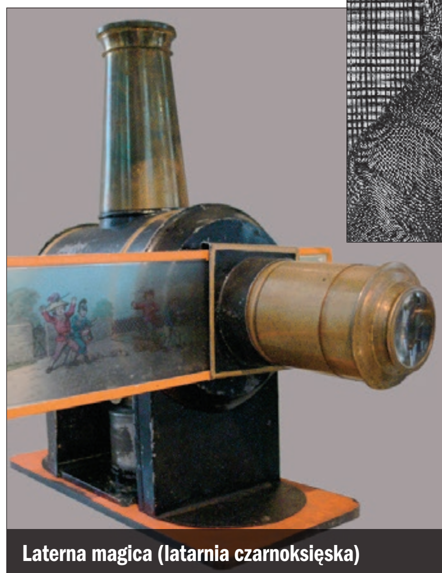
# 1 Animacja: skąd się wzięła i czym jest

**Animacja przeszła długą drogę, ale zawsze najważniejsze w niej jest to, że opiera się na połączonych różnych stadiach rysunku czy obrazu, które razem dają iluzję ruchu. Przeczytajmy, skąd się wzięła animacja, by zrozumieć, na czym polega**

Animacja 3D, której jest poświęcona ta książka, to komputerowe połączenie różnych metod znanych od lat – trochę rysunku, trochę lalkarstwa, trochę podpatrywania świata przyrody, by ruch wyglądał naturalnie. Animacja to technika filmowa, dzięki której ożywia się obrazy i wyświetla się ich ruch.

Dziś animacje są czymś powszednim. Oglądamy seriale animowane ze Spider-Manem, japońskie anime czy kreskówki w stylu rodzimego Reksia. Wielu twórców spędziło godziny w studiach, przed komputerem czy przy stołach kreślarskich, by przygotować dla nas te filmy.





Laterna magica (latarnia czarnoksięska)

Fot. pl.wikipedia.org

Każdy, kto gra w gry komputerowe, docenia znaczenie animacji. Ruch mieczem w Elden Ring, kopnięcie piłki w FIF-ie czy tańce w Fortnicie, które cieszą nasze oczy, są efektem pracy twórców animacji.

Tym, co odróżnia animację od obrazu, jest przede wszystkim właśnie ruch, czyli przesunięcie w czasie. Dziś tworzymy je, nagrywając filmy, jednak animacja powstała jeszcze przed pierwszymi filmami.

## Trochę historii

Próby przedstawienia ruchu podejmowali już ludzie w paleolicie, rysując na ścianach jaskiń zwierzęta tak, by „ożywiało je" migotliwe światło ogniska. Przytacza się też przykłady ze starożytnych Chin, gdzie uzyskiwano ruchome obrazy, wykorzystując specjalne lampy, których elementy poruszały się pod wpływem ciepła. Chińczycy są znani z tworzenia urządzeń, które w Europie zaczęły się pojawiać setki lat później.

W XVII wieku jezuita **Athanasius Kircher**, niemiecki wynalazca i teolog, opisał udoskonaloną przez siebie **latarnię czarnoksięską**, nazywaną po łacinie **laterna magica**. Narzędzie to można raczej porównać do rzutnika pojedynczych slajdów, ale uważa się je za protoplastę technologii filmowej.
Prawdopodobnie nie był on pierwszą osobą, która używała urządzenia podobnego typu. Mówi się, że na dworze króla Zygmunta Augusta jeden z jego czarnoksiężników, znany jako Jan Twardowski, wywołał ducha ukochanej króla, Barbary Radziwiłłówny, korzystając z takiego właśnie pierwotnego projektora.
W 1832 roku Joseph Plateau stworzył urządzenie o nazwie **fenakistiskop**. Jest to zabawka w kształcie koła, na którym były przedstawione kolejne klatki animacji. Wprawiony w ruch obraz odbity od tafli lustra dawał złudzenie ruchu.
Kilka lat później **William George Horner** udoskonalił to urządzenie, tworząc **zoetrop**. Był to cylinder z otworami, przez które spoglądało się do środka. Tam przedstawione były kolejne klatki animacji, które dzięki efektowi stroboskopowemu dawały efekt jakby animacji, gdy urządzenie wprawiło się w ruch.


Zoetrop

Fot. pl.wikipedia.org


Fenakistiskop

Fot. pl.wikipedia.org

Z końcem XIX i początkiem XX wieku zaczęto tworzyć filmy, które stopniowo doprowadziły do powstania animacji, jaką znamy dziś. XX wiek to powstanie Walt Disney Pictures, Warner Brothers czy Hanna-Barbera. Produkcje Disneya stały się popularne w Japonii po drugiej wojnie światowej. To właśnie dzięki nim w Kraju Kwitnącej Wiśni rozwinęło się **anime**. W Japonii słowo


Kadr z filmu anime Księżniczka Mononoke


Kadr z filmu Toy Story 2

to określa po prostu właśnie film animowany.
Na przełomie lat siedemdziesiątych i osiemdziesiątych XX wieku nastąpił rozwój grafiki trójwymiarowej. Pierwszym filmem w pełni animowanym trójwymiarowo było **Toy Story**.


Kadr z filmu Za króla Krakusa

Fot. www.sfp.org.pl

## Animacje w grach

Grafika w grach komputerowych na początku nie cechowała się szczegółowością, ale dzięki rozwojowi technologii gry stawały się coraz bardziej zaawansowane. Pierwsze gry wyglądały jak poruszające się zbiory pikseli wyświetlane na ekranie. Potem twórcy gier przeszli do animowanych spritów, czyli obrazków przedstawiających kolejne klatki animacji. Wraz z powstaniem lepszych komputerów i narzędzi pojawiła się możliwość ingerowania w to, jak się rusza postać, w czasie rzeczywistym. Pojawiła się między innymi **fizyka ragdoll**. To symulacja fizyczna, która pozwoliła zastąpić ruch ciała animowany ręcznie na generowany komputerowo.


Kadr z filmu Piękna Lukanida

Fot. pl.wikipedia.org

## Polska animacja

Polacy też mają wkład w rozwój filmów animowanych. **Władysław Starewicz**, filmowiec narodowości polskiej, który pracował na Litwie, w Rosji i we Francji, stworzył w 1912 roku pierwszy film animowany techniką lalkarską, który wszedł do dystrybucji kinowej. Film ten, zatytuowany **Piękna Lukanida**, nawiązuje do greckiego mitu o Helenie i Parysie, a najciekawsze jest to, że główne role odgrywają w nim żuki jelonki.
Po drugiej wojnie światowej nastąpił w Polsce rozwój filmów animowanych. Pionierem był **Zenon Wasilewski**. Za jego największe osiągnięcie uznaje się film **Za króla Krakusa** z 1947 roku, pierwszy profesjonalny polski film animowany dla dzieci.
W 1957 roku **Jan Lenica** i **Walerian Borowczyk** stworzyli film **Był sobie raz...**. Jest to historia o plamie tuszu, która wyrusza w świat. Animacja została stworzona z wycinanek. W 1958 roku ci sami twórcy nakręcili eksperymentalny film animowany **Dom**, surrealistyczną historię kamienicy, która jest świadkiem wielu różnych zdarzeń.
Jedna z najbardziej znanych polskich wytwórni filmów animowanych to łódzkie studio **Se-Ma-For** (Studio Małych Form Filmowych). Zaczęło ono działać w 1947 roku. Niestety, w 2018 studio zostało zamknięte.


Kadr z filmu Miś Uszatek

Fot. wmeritum.pl

Stworzono w nim takie klasyki, jak:
- Miś Uszatek,
- Piotruś i wilk,
- Przygody Misia Colargola,
- Muminki.

Inną znaną polską wytwórnią jest **Studio Filmów Rysunkowych w Bielsku-Białej**. Jego znane filmy to:
- Sceny z życia smoków,
- Reksio,
- Bolek i Lolek,
- Wyprawa profesora Gąbki.




Współcześnie w Polsce najprężniej działa studio **Platige Image**. Związany z nim **Tomasz Bagiński** stworzył filmy **Katedra** i **Legendy polskie**.

## Techniki tworzenia animacji

Możemy wyróżnić kilka technik tworzenia filmów animowanych:
- **Rysunkowa** – najbardziej popularna, kolejne klatki animacji są rysowane czy to na papierze, czy w programie graficznym; przykładem mogą być filmy animowane studia Ghibli.
- **Lalkowa** – twórcy kreują świat przedstawiony za pomocą lalek, figurek bądź innych obiektów; za przykład mogą posłużyć Muminki ze studia Se-Ma-For.
- **Animacja rotoskopowa** – jest połączeniem animacji rysunkowej i filmu aktorskiego; twórcy najpierw nagrywają aktorów, a potem na podstawie kolejnych klatek rysują ruch postaci. Dobrym przykładem jest film **Władca Pierścieni** (1978) **Ralpha Bakshiego** oraz pierwsza gra z serii **Prince of Persia**, gdzie twórca nagrywał ruchy swojego brata, by przełożyć je na animację postaci.
- **Animacja 3D** – filmy tworzone za pomocą programów do grafiki trójwymiarowej; filmy z wytwórni **Disney Pixar** w większości tworzone są tą techniką.
- **Animacja motion capture** – połączenie animacji rotoskopowej oraz trójwymiarowej. Filmuje się aktorów i za pomocą specjalnych algorytmów ich ruch przekłada się na ruch postaci 3D.



Kadr z animacji Muminki studia Se-Ma-For

Fot. pl.pinterest.com



Kadr z filmu Władca Pierścieni Ralpha Bakshiego

Fot. www.councilofelrond.com

# ZASADY ANIMACJI

Najsławniejszym twórcą animacji jest Walt Disney. W stworzonej przez niego wytwórni Walt Disney Productions (dziś The Walt Disney Company) opracowano 12 zasad animacji.


Fot. pl.wikipedia.org

**1 Zgniatanie oraz rozciąganie**
Zasada stworzona w celu przekazania dynamiki ruchu. Na każdy obiekt oddziałują siły, które na różne sposoby go odkształcają. Najbardziej klasycznym przykładem jest animacja odbijanej piłki – piłka spadając, odkształca się z tej strony, w którą się porusza, przy zetknięciu z podłożem spłaszcza się, a przy odbiciu znów się wydłuża. Innym przykładem może być klasyczna animacja – uderzona młotkiem głowa postaci wbija się w ciało, by zaraz wrócić na swoje miejsce.

**2 Wyprzedzanie**
Zadaniem tej zasady jest przygotowanie odbiorcy na ruch, który zaraz się zacznie. Przykładem może być wystraszony kotek, który chce zacząć biec, ale podłoga jest tak gładka, że w pierwszych sekundach biegnie w miejscu. To jest właśnie wyprzedzanie – wiemy, że kotek zaraz wystrzeli jak z procy.

**3 Inscenizacja**
Ta zasada ma skierować uwagę widza w konkretne miejsce, poinformować, że dane ujęcie jest teraz ważne. Służą do tego różnego rodzaju reakcje, linie, odpowiednie światło lub kadrowanie.

**4 Rysowanie według klatek kluczowych oraz progresywne** to dwa rodzaje tworzenia ruchu.

- **Tworzenie według klatek kluczowych** – metoda ta polega na stworzeniu wybranych klatek (stanów) animacji, a następnie dodaniu przejść między nimi. Jest bardzo popularna w grafice komputerowej, gdzie na przykład nasz model ma przygotowany szkielet i animator może ustalić, że w danej sekundzie w danej klatce ręka ma być w miejscu A, w drugiej klatce w miejscu B i to jeszcze jako pięść.
- **Rysowanie progresywne** – metoda, w której kolejno rysuje się klatki animacji, nie mając żadnych wcześniejszych stanów, do których się dąży. Bardzo popularna w scenach akcyjnych. Ma jednak wady, gdyż korzystając z niej, trudno zachować proporcje. Obecne narzędzia komputerowe wyeliminowały ten problem.

**5 Nakładanie się i podążanie za akcją**
Zasada pozwalająca na spotęgowanie wrażenia ruchu. Składa się na nią kilka elementów.

- Podążanie to pewnego rodzaju bezwład. Gdy na przykład postać nagle zostanie zatrzymana, to nogi już się nie ruszają, ale tułów podąża dalej; powrót do równowagi zajmuje postaci jakiś czas.
- Nakładanie się to założenie, że poszczególne elementy postaci powinny się poruszać w innych momentach. W momencie gdy rusza się ręka, nie rusza się głowa itp.

- Ciągnięcie to zasada mówiąca, że gdy jakieś elementy są w relacji, to ruch kolejnego jest opóźniony względem poprzedniego. Przykładem są animacje, w których ciało postaci odjeżdża za kadr, a głowa jeszcze w nim pozostaje i dopiero po chwili podąża za resztą.

**6 Rozpędzanie i zwalnianie**
Zasada korespondująca z nakładaniem i wyprzedzeniem. Masywny obiekt będzie dłużej się rozpędzał niż mniejszy i tak samo dłużej będzie się zatrzymywał.

**7 Łuki**
W naturze większość obiektów żywych porusza się po łuku. Zastosowanie tej zasady w animacji dodaje realizmu. Tu ważne też jest, co się porusza. Trasy ruchu obiektów organicznych, takich jak ludzie czy zwierzęta, będą bardziej wygięte, natomiast elementy nieożywione, maszyny, będą się raczej poruszać po prostych torach.

**8 Akcja drugoplanowa**
Dodanie drugoplanowej animacji urealnia ruch i uwydatnia główne elementy animacji. Jako przykład może posłużyć podskakująca czapeczka na głowie chodzącej postaci czy postać drugoplanowa poruszająca się wraz z główną postacią.

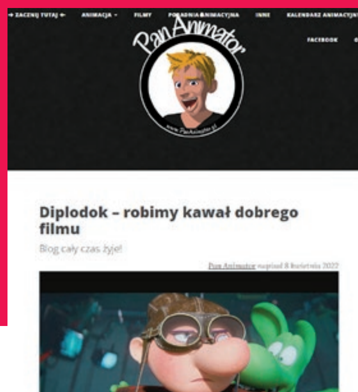
**9 Taktowanie**
Czas ruchu powinien odnosić się zarówno do praw fizyki, jak i do historii opowiadanej przez animację. Jeśli nie mamy do czynienia z przypadkiem szczególnym, zamierzonym przez twórcę w celu osiągnięcia określonego efektu, taktowanie jest poprawne wtedy, gdy obiekty pojawiają się zgodnie z prawami fizyki (masa określa prędkość i wpływ obiektu na pozostałe elementy sceny). Taktowanie teatralne jest związane z tą dziedziną sztuki. Polega na odpowiednim wyważeniu czasu ruchu, który prowadzi do komicznych lub dramatycznych efektów.

**10 Wyolbrzymienie**
Konwencja stosowana w animacji – atrakcyjne dla widza jest przedstawianie przejaskrawionych ruchów, na przykład postać, która jest zdenerwowana, nadyma się i rozszerza, by niemal dosłownie wybuchnąć z gniewu.

**11 Rysunek przestrzenny**
Odwzorowanie trójwymiarowości. Kładzie nacisk na proporcje i kształt postaci w danej chwili ruchu.

**12 Urok**
To, co w animacji odpowiada charyzmie aktora. Postać nie musi być miła, ale ważne jest, by była wiarygodna, charakterystyczna i interesująca.

Wiele z tych zasad może się wydawać nieadekwatnych w animacjach realistycznych, ale do dzisiaj są one stosowane nie tylko w filmach animowanych, ale również w grach, gdy twórcy chcą podkreślić dynamizm postaci.
Ciekawie zaprezentowane przykłady i opisy można znaleźć na stronie:
**pananimator.pl/animacja/12-zasad-animacji**



# 2 Warte uwagi narzędzia do animacji



Na DVD znajdziemy programy przedstawione w tym rozdziale

**Zanim w kolejnych rozdziałach poznamy dokładnie głównego bohatera tej książki – najbardziej rozbudowany i znany program do tworzenia animacji, czyli Blender – przeczytajmy, jakie mamy do wyboru darmowe narzędzia, które mogą wprowadzić nas w świat animacji**

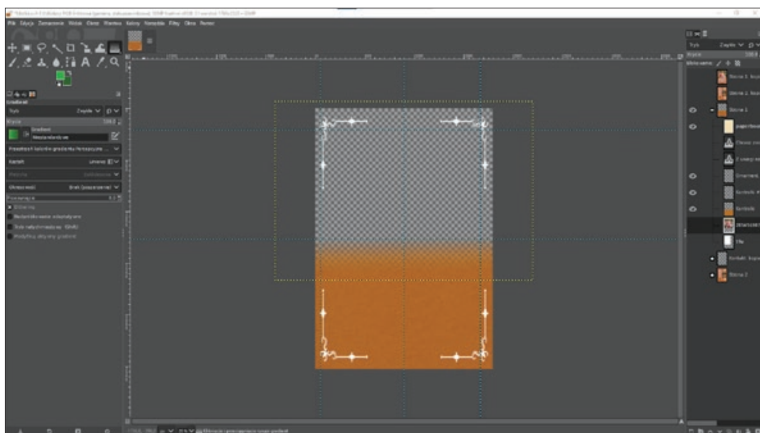
## Darmowe narzędzia do nauki

Doświadczenie pokazuje, że ważne, ciekawe, zapadające w pamięć oraz innowacyjne projekty często tworzą amatorzy i zapaleńcy, którzy, nieraz po nocach, ślęczą nad kartkami papieru, rolkami filmu czy lalkami. Nie zawsze przy tym od razu udaje się osiągnąć zamierzony efekt, z reguły wymaga to wielu prób i poprawek. Dziś mamy na szczęście znacznie ułatwione zadanie. Powszechna dostępność narzędzi – także darmowych, tutoriali czy szkoleń – ułatwia poszukiwania najlepszych rozwiązań i doskonalenie swoich umiejętności. Wiele firm tworzy też na swój użytek narzędzia, które potem szeroko udostępnia, by amatorzy i pasjonaci mogli stosować je w swoich projektach. Poznajmy najlepsze darmowe programy do nauki tworzenia animacji.

# GIMP



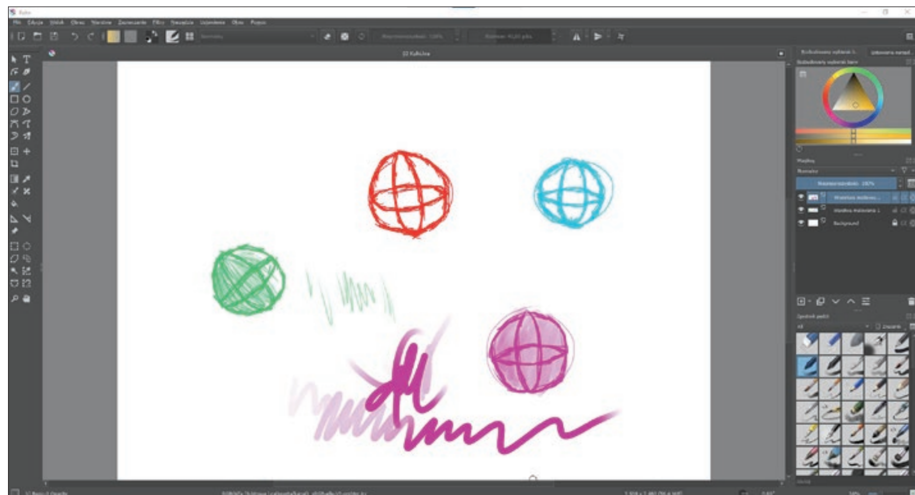
**GIMP** (DVD-KOD: 006) to najbardziej znany darmowy edytor graficzny. To program open source do grafiki rastrowej. Program ten służy przede wszystkim do obróbki fotografii. Jednak jego możliwości na tym się nie kończą. Dzięki temu, że GIMP ma otwarty kod źródłowy, użytkownicy tego programu mogą samodzielnie dodawać rozszerzenia. Sam w sobie GIMP ma możliwość zapisywania plików w formacie GIF, co pozwala tworzyć w nim proste animacje, zapisując klatki na kolejnych warstwach.



# Krita



**Krita** (DVD-KOD: 007) to popularny i darmowy program (choć na Steamie trzeba go kupić) do malowania i tworzenia grafiki rastrowej od podstaw. Umożliwia tworzenie animacji. Zawiera narzędzie, które pozwala na podgląd tworzonej animacji, przewijanie czy zarządzanie stanem klatek. Pozwala nawet na dodawanie dźwięku do animacji.



# Synfig



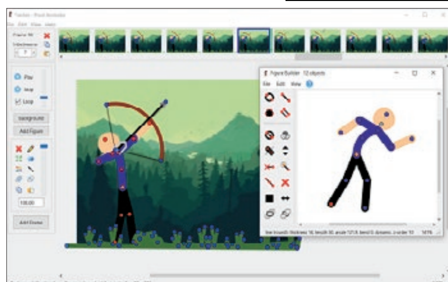
**Synfig** (DVD-KOD: 016) to program open source, pozwala operować też na grafice wektorowej. Nie trzeba tworzyć każdej klatki osobno, ale można zająć się transformacją wybranych elementów sceny w jej przestrzeni. Na przykład gdy mamy scenę, w której postać podnosi rękę, nie musimy rysować tej ręki klatka po klatce. Wystarczy, że podamy, w której klatce i w którym miejscu ma się znajdować ręka, a program dokona przeliczeń tworzących animację. Synfig pozwala na animację na kościach (więcej o animacji na kościach – w dalszej części książki).

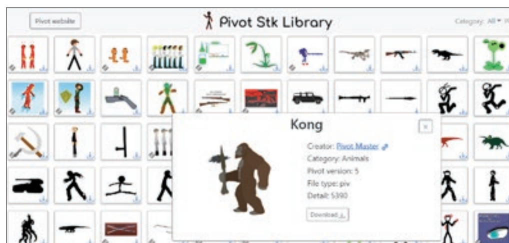



# Pivot Animator




**Pivot Animator** (DVD-KOD: 012/013) to ciekawy, prosty program graficzny do tworzenia animacji 2D, który opiera swoje działanie na wektorach. Służy przede wszystkim do tworzenia animacji ludków – patyczaków. Zawiera rozbudowany edytor do tworzenia i animacji tych prostych postaci. Korzysta z wektorowych figur (obiektów), które są wyposażone w uchwyty (węzły) i które można zaimportować lub utworzyć we wbudowanym edytorze figur. Na stronie **pivotanimator.net** znajdziemy bibliotekę darmowych postaci do pobrania.

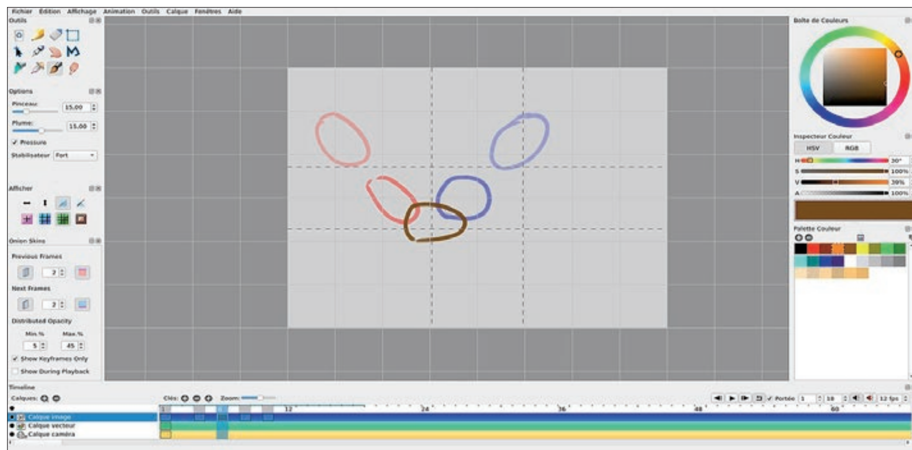




# Pencil 2D



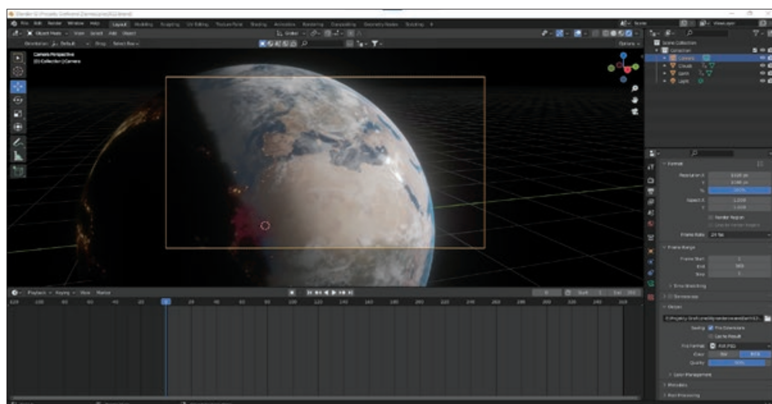
**Pencil 2D** (DVD-KOD: 011) – minimalistyczny, ale bardzo dobry program do tworzenia animacji 2D. Pozwala na tworzenie zarówno elementów rastrowych, jak i wektorowych, a także dodawać dźwięki. Jest bardzo popularny. Na stronie **pencil2d.org** znajdziemy tutoriale i przykładowe projekty zrealizowane w tym programie.



# Blender



**Blender** (DVD-KOD: 002/003 PORTABLE) to jeden z najbardziej rozpoznawalnych i popularnych programów do grafiki i animacji 3D, w nowszych wersjach pozwala również na tworzenie projektów 2D. Rozwijany przez Blender Foundation, jest dostępny tak jak GIMP czy Krita na systemy Linuxa, macOS i Windows. Ma bardzo duże możliwości, począwszy od animowania ruchu obiektów, przez kości i grupy wierzchołków aż po transformacje różnych elementów za pomocą węzłów. To naprawdę potężne narzędzie – w dalszej części będziemy mieli możliwość zapoznania się z nim.



# FlipaClip



**FlipaClip** to jedyna w tym zestawieniu aplikacja na platformy mobilne – ma wersje na Androida i iOS. Pozwala na tworzenie animacji 2D klatka po klatce. Można w niej układać proste animacje (wskazówkę krok po kroku, jak to robić, znajdziemy w pliku PDF w KŚ+ – **ksplus.pl** – oraz na płycie dołączonej do książki). We FlipaClip można rysować palcem po ekranie dotykowym, ale rysik jest z reguły lepszym rozwiązaniem. Program w wersji z reklamami jest dostępny za darmo.




# Narzędzia dla zaawansowanych

Oprócz darmowych narzędzi wiele firm korzysta z zaawansowanych, komercyjnych programów. Ich licencje są często bardzo drogie, ale możliwości, jakie dają firmie, są zazwyczaj tego warte. Zdarza się, że mają one darmowe odpowiedniki lub darmowe wersje.

# Toonz



To program do grafiki 2D, który pozwala zarówno na tworzenie grafiki rastrowej, jak i na zabawę wektorami czy dodawaniem kości. Obecnie jest dostępny w dwóch wersjach. **Toonz Premium** jest płatny. Ale w 2016 roku wydano **OpenToonz** (DVD-KOD: 010), czyli otwartoźródłową, darmową wersję.
Tym narzędziem posługuje się znane **Studio Ghibli**. Pracują w nim od 1995 roku, a pierwszym filmem stworzonym za pomocą tego narzędzia była **Księżniczka Mononoke** (1997). Innym znanym studiem, które korzysta z tego narzędzia, jest **Rough Draft** znane z serii **Futurama**.
**Monolith Productions** korzystało z tego narzędzia, gdy tworzyło grę **Claw** (1997), w Polsce znaną jako **Kapitan Pazur**.









# Toon Boom

**Toon Boom** (DVD-KOD: 018/019 ADVANCED/ESSENTIAL) to rodzina programów do tworzenia grafiki. Oferuje bardzo szeroki zakres możliwości. Jest to bardzo popularne narzędzie. Korzystają z niego zarówno studia **Disneya**, jak i **WarnerBros**. Programy Toon Boom są stosowane na przykład do tworzenia animacji dla dorosłych, takich jak **Rick and Morty** czy **Harley Quinn**.





# Adobe

Adobe jest właścicielem takich marek, jak **Photoshop**, **Illustrator**, **After Effects** czy **Animate**. To ostatnie narzędzie wcześniej nazywało się **Flash**, jednak technologia Flash przestała być wspierana, więc zmieniono nazwę programu i sposób tworzenia animacji. Animate daje możliwość tworzenia grafiki wektorowej, a następnie animowania. Wiele gier czy filmów powstało z wykorzystaniem programów Adobe. Często są to elementy stworzone w jednym programie i połączone w innym. Przykładem studia, które korzysta z narzędzi Adobe, jest popularnonaukowy kanał youtubowy **Kurzgesagt** (18,9 mln subskrybentów), specjalizujący się w minimalistycznych animowanych treściach edukacyjnych.




# AutoDesk




Firma **Autodesk** specjalizuje się w oprogramowaniu do wizualizacji, modelowania 3D i symulacji. Najpopularniejszymi narzędziami z tej rodziny są **Maya** oraz **3ds Max**. W Mai są tworzone odcinki **South Parku**. Oba te programy służą do modelowania i animacji 3D. Często korzysta się z nich podczas produkcji gier.



# 3 Wprowadzenie do Blendera



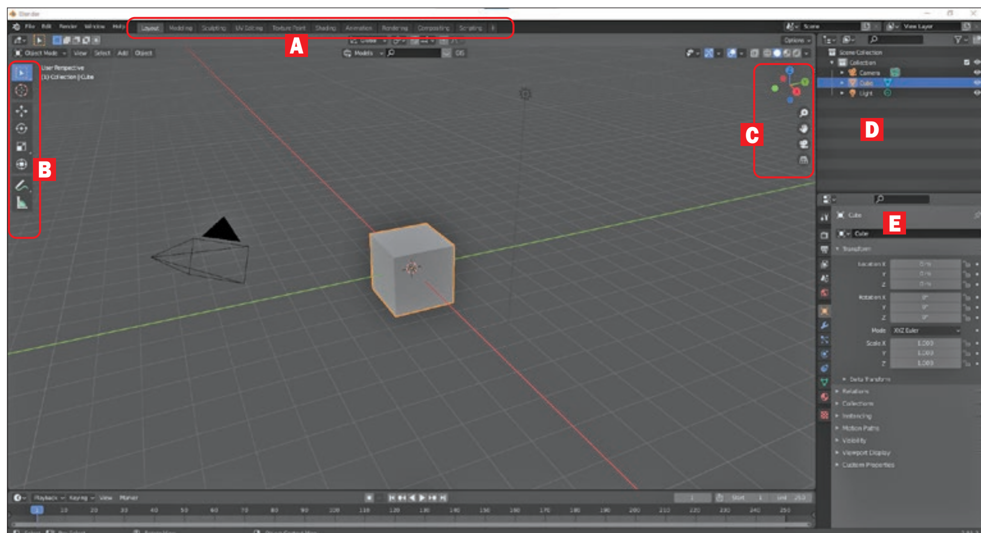

**Wiemy już z poprzednich rozdziałów, co to są animacje, i poznaliśmy najłatwiejsze narzędzia do ich tworzenia. W tym rozdziale zapoznamy się z najlepszym profesjonalnym darmowym programem do animacji. Oto Blender**

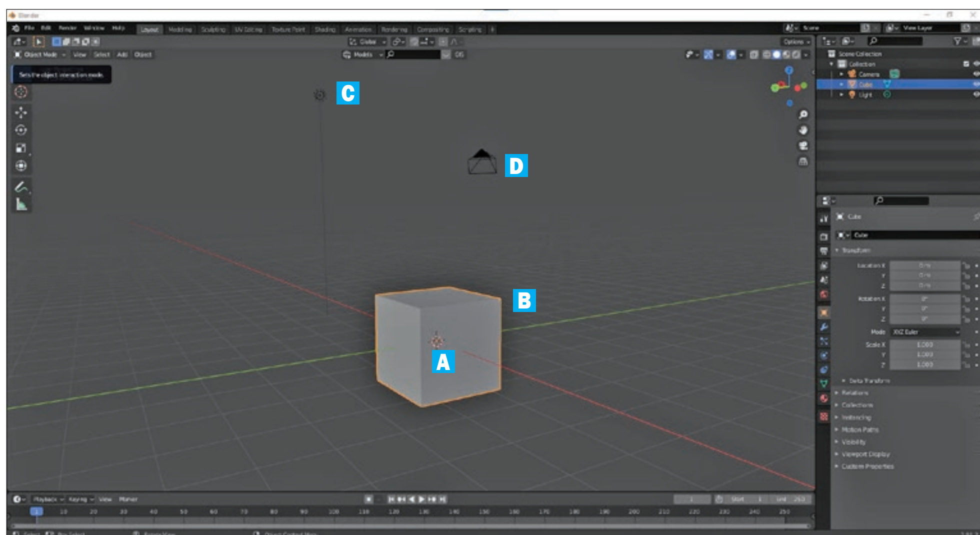
## Okno Blendera

Po włączeniu **Blendera** (**DVD-KOD: 002**) można zauważyć kilka ważnych elementów. Znajdujemy się w zakładce **Layout**, odpowiada ona za tryb **Object Mode**. Wyszczególnimy tu kilka podstawowych elementów:

**A** **Zakładki** – pozwalają przechodzić między oknami Blendera. Najważniejsze dla nas to:

- **Layout** – pozwala na przemieszczanie obiektów, dodawanie modeli, ustawianie kamery czy światła (**Object Mode**),






- **Modeling** – pozwala na modelowanie pojedynczego obiektu (**Edit Mode**),
- **UV Editing** – pozwala na rozkładanie siatki UV i nakładanie tekstur,
- **Shading** – pozwala na tworzenie shaderów,
- **Animation** – pozwala na tworzenie animacji,
- **Rendering** – pozwala na renderowanie obrazów czy klipów animacji.

B **Przybornik** – z niego wybieramy narzędzia do edycji obiektów na scenie.

C **Sidebar** – tu znajdują się narzędzia do poruszania się po scenie oraz zakładki do edycji właściwości poszczególnych elementów.

D **Outliner** – nazywany też drzewkiem obiektów, tutaj wypisane są wszystkie obiekty znajdujące się na scenie.

E **Properties** – to okno pozwala na ustawienia takich właściwości, jak parametry renderowania, dodawanie materiałów lub modyfikatorów.

Na scenie mamy widoczne:

A **Pivot point/3D Cursor** – punkt środka; gdy dodaje się nowy element, jest on dodawany właśnie w tym punkcie. Względem niego mogą być dokonywane zmiany na wielu obiektach, takie jak obrót czy skalowanie.

B **Cube** – bryła, która pojawia się zawsze w nowym projekcie.

C **Light** – obiekt światła, odpowiada za rodzaj, kolor czy moc światła na scenie.

D **Camera** – obiekt kamery służący do renderowania obrazów i animacji.

# Poruszanie się po scenie Blendera, skróty widoku

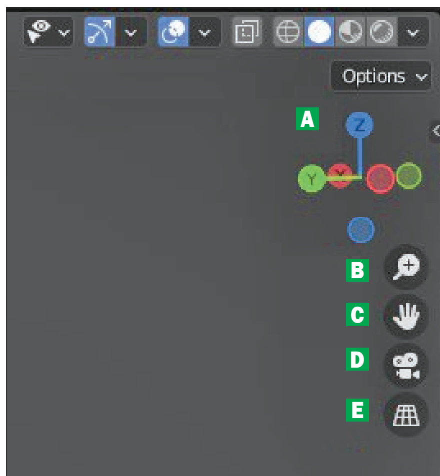
Pierwszą rzeczą, jakiej należy się nauczyć, pracując w programie do grafiki, jest sposób, w jaki można się po nim poruszać. Tutaj robimy to za pomocą myszy.

- **Przytrzymując wciśnięte kółko myszy i poruszając nią** – obracamy się względem środka sceny lub obiektu,
- **Wciskając** shift **i przytrzymując wciśnięte kółko myszy i poruszając nią** – uzyskujemy przesuwanie 2D, czyli przesunięcie góra-dół i prawo-lewo,
- **Kręcąc kółkiem myszy lub wciskając klawisz** ctrl **i kółko myszy oraz poruszając myszą** – przybliżamy się i oddalamy.

**Widok perspektywiczny** – to taki widok, jaki widzą ludzkie oczy, jest w nim głębia.
**Widok ortograficzny** – widok płaski, nie ma w nim głębi, jest przydatny, gdy pracujemy nad wyrównywaniem elementów modelu.

W ostatnich wersjach Blendera powyższe operacje można wykonywać, klikając na odpowiednie elementy. Wystarczy na nie kliknąć lub przytrzymać i poruszać myszą:

**A** Klikanie na kolorowe kulki na pivocie pozwala obserwować modele z każdej strony. Dodatkowo po najechaniu, gdy pojawi się białe koło wokół, można po kliknięciu i przytrzymaniu przycisku myszy obracać obiekt tak jak po wciśnięciu kółka myszy.




**B** Po kliknięciu na tę ikonę można, poruszając myszą, przybliżać i oddalać widok.
**C** Ten przycisk pozwala na przesunięcia w płaszczyźnie ekranu.
**D** Ta ikona pozwala na przełączanie się pomiędzy widokiem a kamerą.
**E** Pozwala na przełączenie się pomiędzy widokiem ortograficznym i perspektywicznym.

# Manipulacje na obiektach

Przy pracy w trzech wymiarach poruszamy się po układzie współrzędnych.

Mamy trzy osie:
- **czerwoną oś X** – czyli ruch prawo-lewo,
- **zieloną oś Y** – czyli ruch przód-tył,
- **niebieską oś Z** – czyli ruch góra-dół.

Klikając lewym przyciskiem myszy na obiekt, zaznaczymy go – będzie miał pomarańczowy kontur.
Teraz obiekt można poddać transformacjom.
Mamy trzy transformacje:
- **przesunięcie** – położenie obiektu na scenie,
- **rotacja** – obrót obiektu,
- **skalowanie** – zmiany jego wielkości.

Zaznaczony obiekt możemy transformować za pomocą przycisków na klawiaturze:
- G (grab) – przesunięcie,
- R (rotate) – rotacja,
- S (scale) – skalowanie.

Możemy ograniczyć się do wybranej osi. Gdy wciśniemy na przykład S (scale), to naciskając na klawiaturze Z, ograniczymy się do osi Z, a naciskając shift+Z, wykluczymy ją.

## Przybornik

Oprócz klawiszy można także użyć przybornika znajdującego się po lewej stronie okna.

**A Zaznaczanie** – przełącza nas w tryb zaznaczania.
**B Ustawienie kursora 3D** – pozwala na ustawienie Cursora 3D w wybranym przez nas miejscu.
**C Przesunięcie**.
**D Rotacja**.
**E Skalowanie**.
**F Transformacje** – włącza widok wszystkich transformacji naraz.

Po wybraniu narzędzia możemy kliknąć na któryś z kolorowych elementów i poruszyć myszą, co pozwali na wykonanie odpowiedniej transformacji.




# Modelowanie

## Mesh, vertex, edge i face – czym jest model 3D i jak widzi go komputer

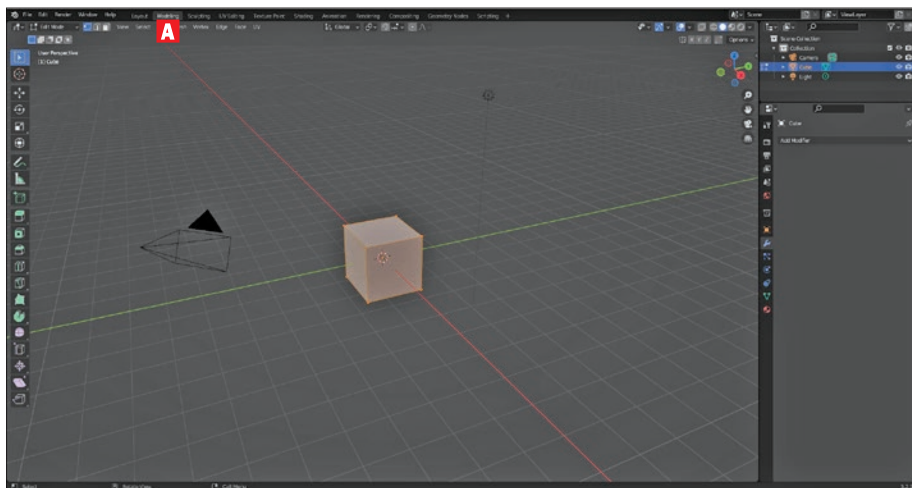
Matematycy i programiści, tworząc wizualizacje 3D, korzystali z różnych sposobów. Jednym z najbardziej uniwersalnych okazała się siatka punktów. Każdy taki punkt ma swoje określone miejsce w przestrzeni 3D nazywane zazwyczaj wektorem (x, y, z). Taki punkt w grafice nazywamy wierzchołkiem lub po angielsku **vertexem**. Vertexy łączy się z sobą. Pomiędzy jednym i drugim powstaje wtedy krawędź, czyli **edge**. Kilka krawędzi łączących się w zamkniętą przestrzeń tworzy **poligon** (polygon), zwany też **face**.

Najmniejsze poligony to trójkąty i czworokąty. Graficy 3D dążą do tego, by poligony były właśnie trójkątami lub czworokątami, bo figury złożone z większej liczby krawędzi mogą tworzyć artefakty graficzne, czyli niepożądane, dziwne błędy.

**Mesh** to pojedyncza, osobna siatka punktów. Model może być jednym meshem, ale może również składać się z kilku meshy.

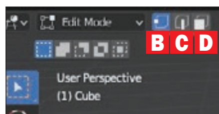
## Transformacje i operacje na siatce punktów

**1** Aby stworzyć nowy projekt w Blenderze, zaznaczamy myszą cube'a na środku sce-



ny lub w drzewku obiektów. By móc przejść do edycji meshy, musimy kliknąć na górze okna na zakładkę **Modeling** **A** (poprzednia strona). Przechodzimy wtedy do **Edit Mode**.

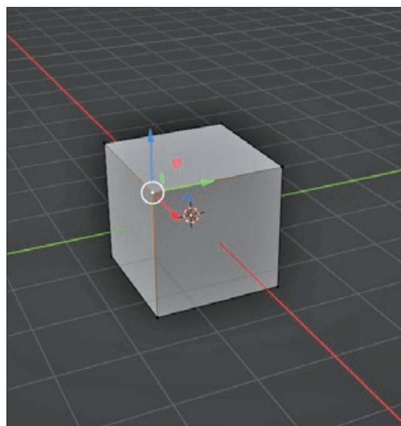
**2** W przyborniku po lewej widać, że pojawiły się nowe narzędzia. Posłużą nam one później. Nad przybornikiem widzimy listę z napisem **Edit Mode**, a po prawej stronie




trzy przyciski. Pozwolą nam one przełączać się między zaznaczaniem wierzchołków (vertexów), krawędzi (edge) i poligonów. Klikamy na pierwszy z nich. Na naszym modelu zobaczymy teraz punkty. Na każdy z nich możemy kliknąć i każdy możemy zaznaczyć.

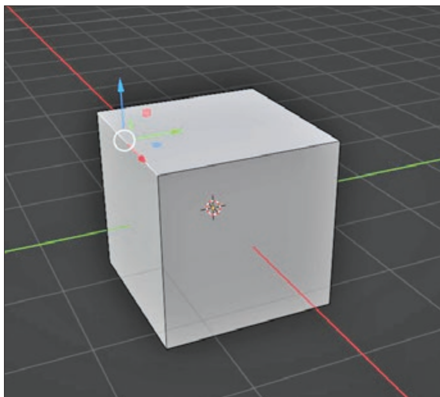
## Vertexy

Klikając na **B**, zaznaczamy vertexy. Podlegają one transformacjom tak jak modele. Pojedynczy vertex może zostać tylko przeniesiony. Przenosimy go, naciskając klawisz G lub klikając na narzędzie w przyborniku. Rotacja i skalowanie na pojedynczym wierzchołku nie mają sensu. Jednak gdy zaznaczymy ich kilka (przytrzymujemy shift i klikamy na wybrane wierzchołki), transformacje mają sens, bo wierzchołki będą między sobą oddziaływać.



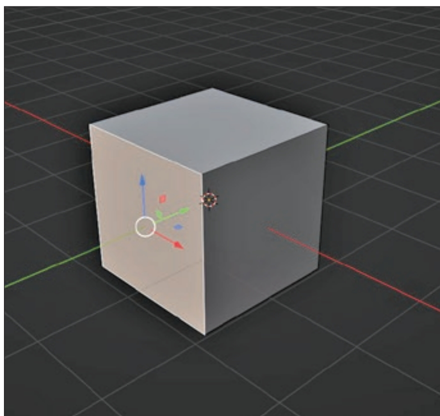
## Edge

Teraz czas poznać krawędzie. Klikamy na drugi przycisk **C**. Znikają vertexy, możemy zaznaczać całe krawędzie. Tutaj działa to dokładnie tak samo jak w wypadku vertexów. Można zauważyć, że jak mamy zaznaczoną krawędź i przełączymy się na vertexy, będziemy mieli podświetlone te, które łączy ta krawędź. Operacje na krawędzi to tak naprawdę praca na dwóch połączonych vertexach jednocześnie.



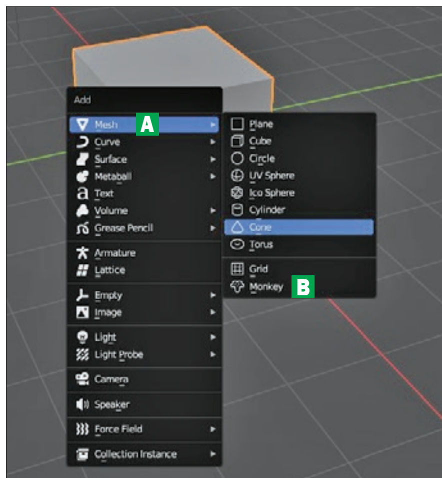
## Poligony

Klikamy na ostatni przycisk **D**. W Blenderze poligony są nazywane face'ami. Tu znowu mamy to samo działanie. Możemy spokojnie przesuwać, obracać czy skalować naszego poligona.



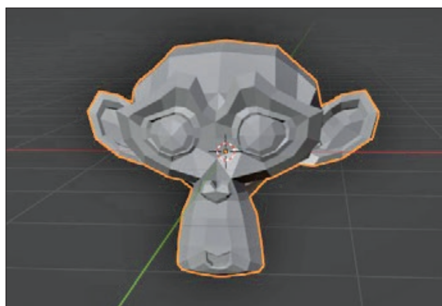
# Dodawanie obiektów

Nie będziemy wprawdzie pracować na samych siatkach, gdyż zajęłoby nam to dużo czasu, warto je jednak poznać – to ważna wiedza, która nam się przyda.

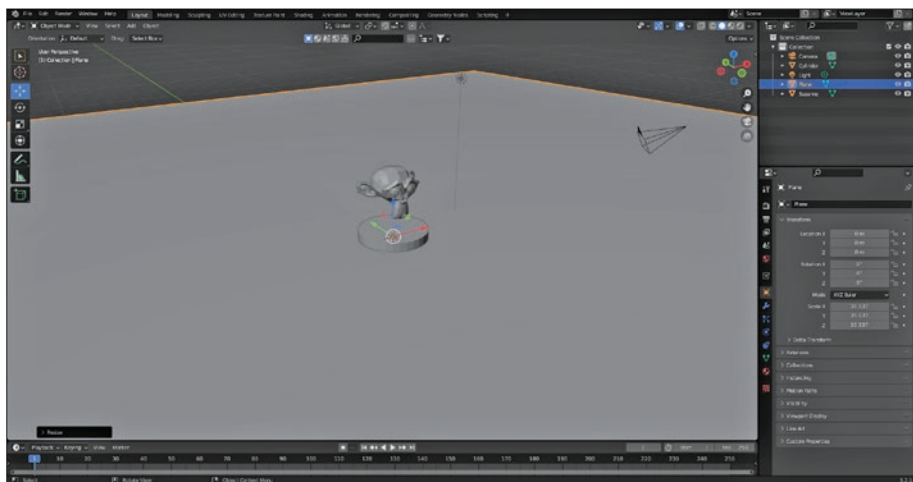



**1** Wróćmy do zakładki **Layout**. W obecnej chwili na scenie mamy tylko cube, światło i kamerę. Aby usunąć obiekt, trzeba na niego kliknąć i nacisnąć klawisz [delete] na klawiaturze. Natomiast by móc dodać nowy obiekt, należy użyć klawiszy [shift]+[A]. Pojawia się wtedy lista z różnymi kategoriami obiektów do dodania. Nas w tej chwili najbardziej interesuje kategoria **Mesh** **A**.

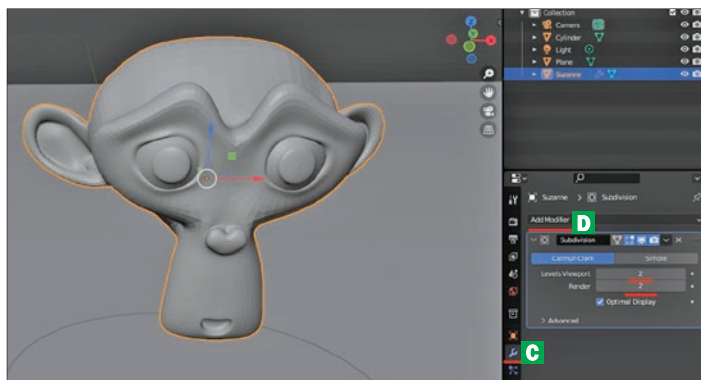
**2** Usuńmy **Cube** z naszej sceny, klikając na niego i naciskając klawisz [delete]. Mamy kilka modeli do wyboru. Dodajmy więc coś specjalnego. Wybierzemy **Monkey** **B**, czyli małpią główkę (nazywa się **Suzanne**), która jest maskotką Blendera.



**3** Dodajmy też jeden **Cylinder**. Spłaszczymy go, tworząc podstawę, nad którą będzie się poruszać główka. Dodajmy także jeden **Plane**, który posłuży jako podłoże.



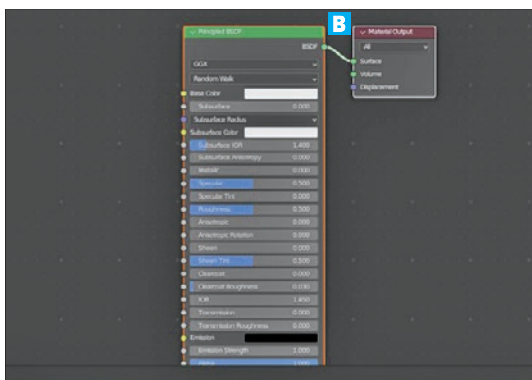
**4** By główka nie była taka kanciasta, przejdziemy do zakładki **Modifier Properties** **C** i dodamy tam modyfikator **Add Modifier**, **Subdivision Surface** **D**. Dzięki temu wygładzimy główkę.

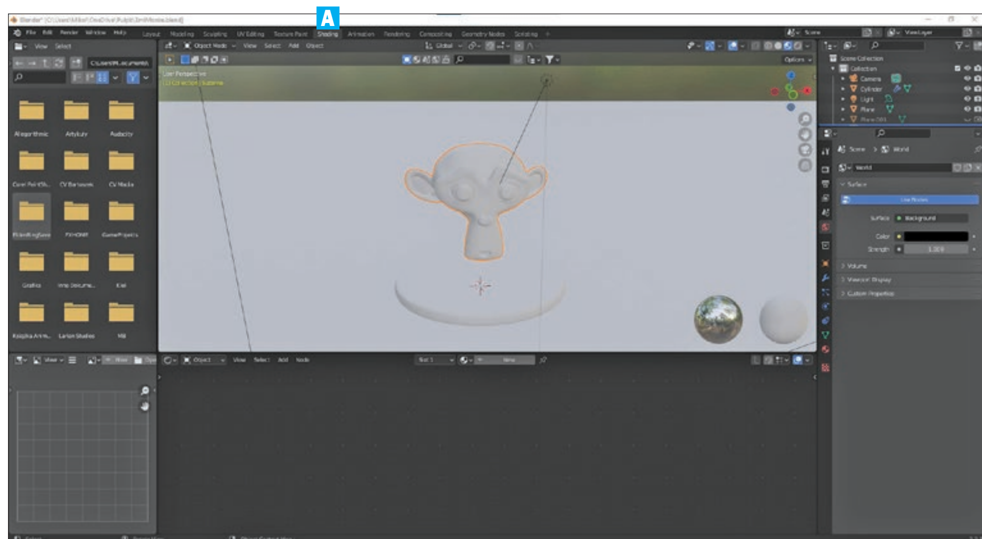


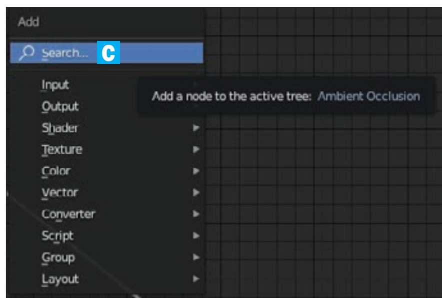

## Shading

**1** Klikamy teraz na naszą małpkę i przechodzimy do zakładki **Shading** **A**. Zajmiemy się tu dodawaniem materiałów – można powiedzieć, że są to kolory i tekstury naszego obiektu (w bardzo dużym uproszczeniu).

**2** By dodać materiał do naszego modelu, klikamy na przycisk **New**. Pojawią nam się wtedy dwa bloczki, zwane też węzłami **B**.



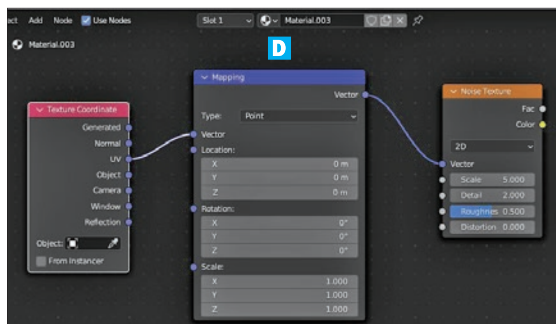



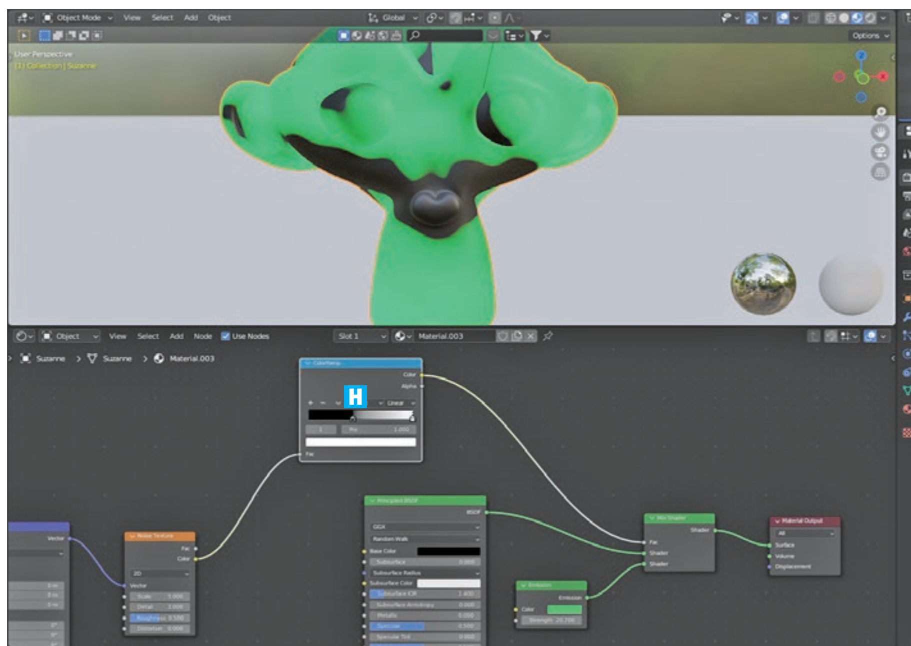



**3** Będziemy tu dodawać kolejne bloczki, które pozwolą nam utworzyć nasz materiał. By dodać nowy bloczek w tym panelu, używamy skrótu shift+A. Pojawia się wtedy lista **C**. W górnej części jest pole **Search...**, które pozwala wyszukać potrzebne nam bloki.

**4** Wyszukajmy kolejne bloczki: **Texture Coordinate**, **Mapping** oraz **Noise Texture**. Noise Texture to tekstura szumu, która generowana jest przez program i możemy ją na bieżąco edytować. Pozostałe bloczki posłużą do odpowiedniego ustawiania tekstury na modelu. Połączmy je tak jak na obrazku **D**.

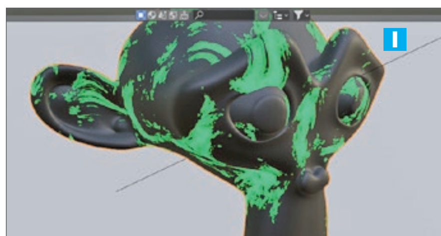
**5** Będziemy chcieli uzyskać efekt świecących plam farby na czarnej powierzchni. Musimy więc zmiksować dwa **shadery**. Nasz pierwszy **Principled BSDF** **E** to taki standardowy shader, który ma wiele różnych właściwości. Nas w tej chwili interesuje **Base Color**, który ustawiamy na czarny **F**. Potem dodajemy bloczek o nazwie **Emission** **G**.






W nim wybieramy odpowiadający nam kolor, a **Strength** ustawiamy na **20**. Emisja pozwala tworzyć iluzję światła generowanego przez obiekt. Jednak teraz nic nie świeci. Musimy te dwa bloczki połączyć, gdyż węzeł **Material Output** może przyjąć tylko jeden bloczek, a to on odpowiada za nakładanie materiału na nasz obiekt. Zastosujemy tu bloczek **Mix Shader**. Do niego podłączymy i **Principled BSDF**, i **Emission**.

**6** Nasza małpka zmieniła kolor, ale w ogóle nie widać plamek i nie emituje światła. Na początek dodamy więc bloczek **ColorRamp**, który pozwoli nam na mieszanie naszych dwóch shaderów na bazie tekstury szumu. Podłączamy do niego bloczek **Noise Texture**,



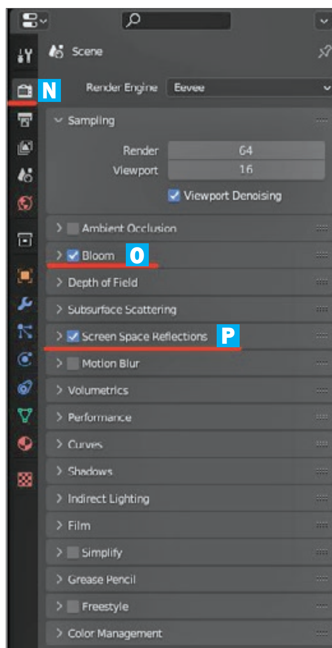



a jego samego podłączamy do **Mix Shader**, do pola **Fac**. Dalej nie widać jednak różnicy, więc przesuńmy czarny suwak w **ColorRamp** **H**. Teraz widać różnicę **I**.

**7** Możemy jeszcze pomanipulować suwakami w **Noise Texture** **J**, by zmienić układ szumu.

**8** Teraz dodajmy nowy materiał do naszego cylindra. W jego wypadku ustawimy tylko właściwości w **Principled BSDF**:

- **Base Color** **K** – dowolny kolor,



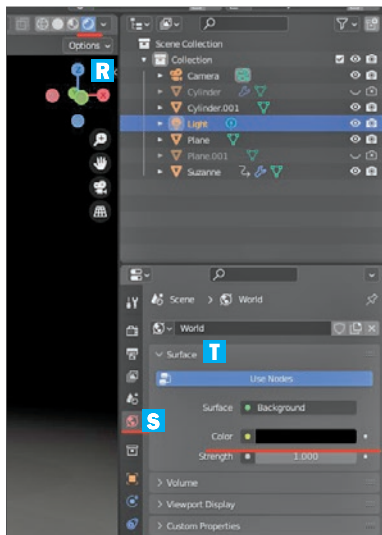

- **Metalic** **L** (odpowiada za metaliczność materiału) – na 1,
- **Roughness** **M** (odpowiada za matowość) – na 0.

Powyższa kombinacja da nam efekt lustra.

**9** Przechodzimy do zakładki **Render Properties** **N** i włączamy dwie właściwości:

- **Bloom** **O** – odpowiada za rozmycie, da efekt światła z naszej małpki,
- **Screen Space Reflection** **P** – odpowiada za odbijanie się elementów w materiałach.

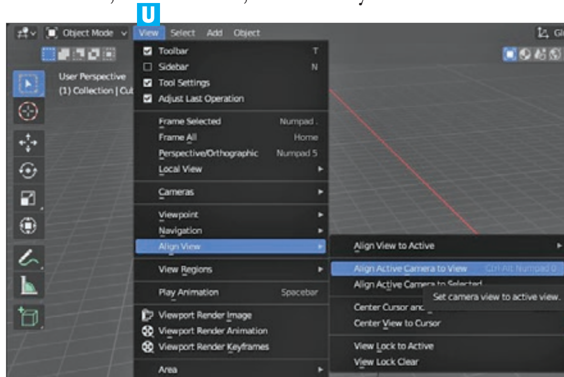
**10** Teraz przechodzimy do **Layout** i przełączamy się na widok renderowania. Słu-




żą do tego kulki w prawym, górnym rogu sceny. Wybieramy tę ostatnią od lewej **R**.

**11** Przechodzimy też do **World Properties** **S** i tam w zakładce **Surface** **T** zmienimy wartość **Color** na czarny. Da nam to efekt czarnej przestrzeni.

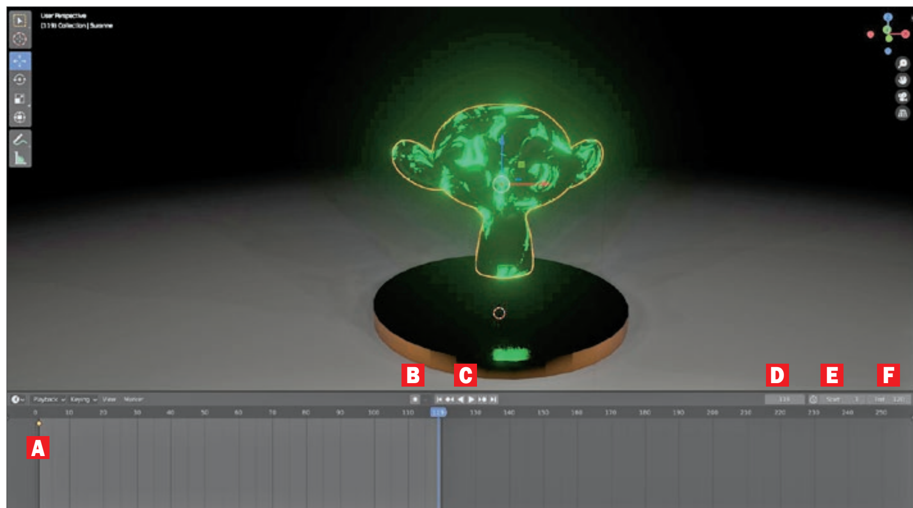
**12** Ostatnia sprawa to ustawić kamerę. Obracamy się tak, jak chcemy wyrenderować naszą małpkę, czyli jak chcemy jej zrobić zdjęcie lub film. Następnie klikamy na **View** **U**, **Align View**, **Align Active Camera to View**. Dzięki temu kamera sama ustawi się w naszej pozycji – nie trzeba jej ręcznie ustawiać. A to, co ona widzi, zostanie wyrenderowane.




# Animacja

## Timeline i keyframe

Omówimy teraz tworzenie animacji. Tworzy się je do filmów, gier czy symulacji. Filmy to wyrenderowane po kolei klatki. Na dole naszej sceny znajduje się panel **Timeline** **A**. Powiększamy go, by był lepiej widoczny.



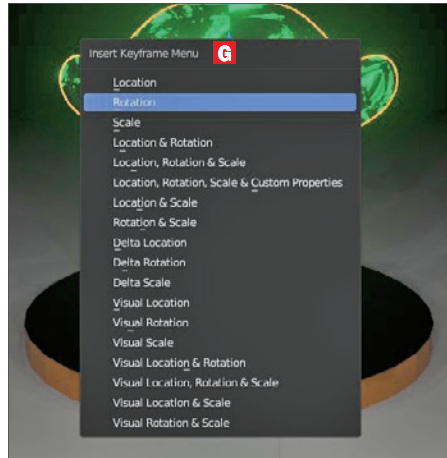
Animacja to praktycznie transformacja w czasie. Ustalamy czas trwania animacji i w odpowiadających nam klatkach ustawiamy nową transformację. Na naszej linii czasu określamy punkt, w którym będą zapisane ustawienia, gdzie ma się w tej chwili znajdować obiekt. Taki punkt nazywamy **keyframe** lub inaczej klatka kluczowa.
Elementy naszej linii czasu to:

- **B** **Auto-frame** – pozwala na automatyczne ustawianie keyframe'ów.
- **C** Przyciski związane z odtwarzaniem w przód i w tył. Trójkącik z beleczką pozwala przejść na koniec lub początek animacji, a trójkącik z rombem – do następnego keyframe'a.
- **D** W tym miejscu możemy wpisać, do jakiej klatki animacji chcemy się przenieść.
- **E** **Start** – tu ustawiamy początkową klatkę animacji.
- **F** **End** – tu ustawiamy końcową klatkę.

## Tworzymy prostą animację

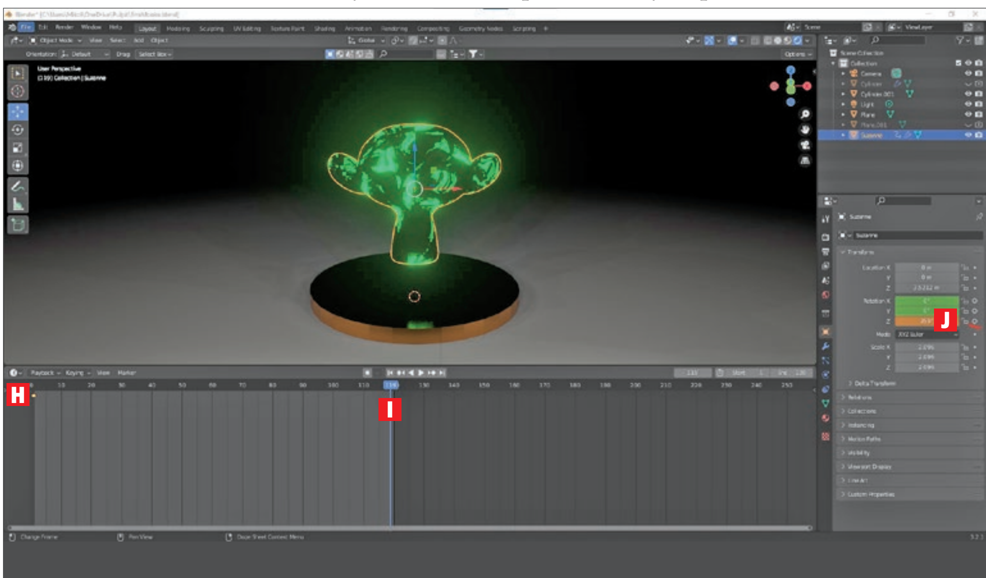
**1** Stworzymy teraz prostą animację. Ustawiamy wartość **end** na 120. Dzięki temu nasza animacja będzie miała 120 klatek.

**2** Teraz możemy stworzyć keyframe. Zaznaczamy naszą małpkę i naciskamy [I] na klawiaturze. Pokaże nam się wtedy lista **G**,



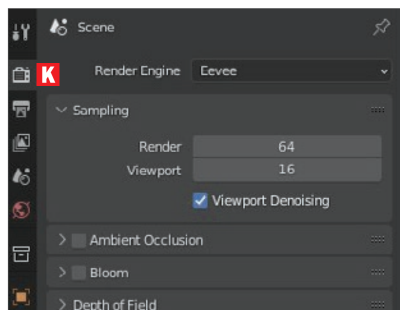

z której możemy wybrać, z jakiego elementu chcemy zrobić keyframe. Klikamy na **Rotation**. Dzięki temu w pierwszej klatce animacji zapamiętamy obecny obrót głowy. Pojawi się mały romb **H**.

**3** Ustawiamy się na linii czasu na 119 klatce **I** i obracamy naszą małpkę o 359 stopni. I znów tworzymy **keyframe**. Możemy to też zrobić w trochę inny sposób. We właściwościach klikamy na zakładkę **Object Properties**. Widzimy tam podświetlone elementy. Ten podświetlony na pomarańczowo **J** to

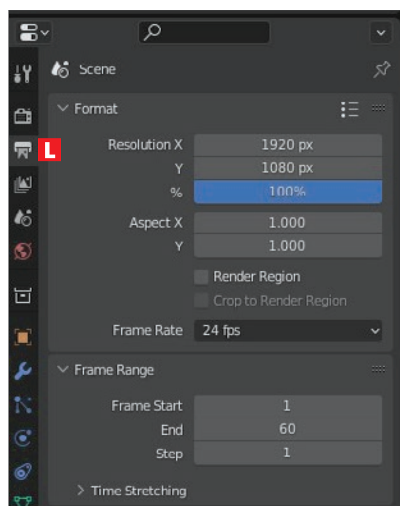


element zmieniony. Obok niego są dwa przyciski. Kłódka pozwala zablokować transformację. Natomiast ten drugi dodaje keyframe.

4 Możemy teraz uruchomić naszą animację. Jednak przed wyrenderowaniem animacji musimy zorganizować pewne elementy. Przechodzimy do zakładki **Render Properties** **K**. Na samej górze mamy element **Render Engine** – tu mamy do wyboru silni-



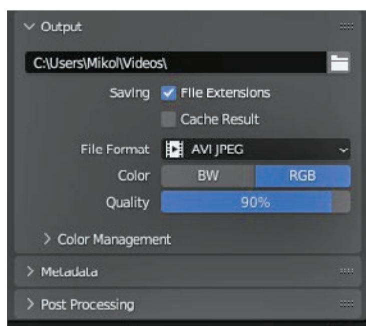

ki renderowania. Z nich interesują nas dwa: **Eevee**, który jest świetny do szybszej pracy, obróbki czy podglądu gotowego efektu, i **Cycles** – silnik, który wspiera dużo więcej elementów, w tym **Raytracing**. My będziemy korzystać z silnika **Eevee**. Warto jednak zapamiętać, że jest alternatywa.




5 Przechodzimy do zakładki **Output Properties** **L**. W zakładce **Dimension** znajdziemy ważne ustawienia.

- **Resolution**. Tutaj ustawiamy rozdzielczość naszego obrazu. Standardowa wartość to 1920 px na 1080 px. Jest to rozdzielczość **HD**. Im większa rozdzielczość, tym dłuższy czas renderowania. Rozdzielczość **4K** to 4096 px na 2304 px.
- **Frame Rate**. Tutaj ustawiamy liczbę klatek na sekundę w animacji. Standardowo ustawiony jest rate filmowy, czyli 24 klatki na sekundę. Po wyrenderowaniu nasza 120-klatkowa animacja będzie trwała prawie 6 sekund. Możemy ustawić inne wartości. Zalecane jest 30 klatek na sekundę. 30 i 60 klatek to standard gier.

6 Teraz zejdziemy na dół do **Output**. Ten element jest bardzo ważny. Klikając na przycisk z ikoną folderu, możemy ustalić lokalizację, gdzie będą zapisywane nasze animacje (domyślnie zostaną wyrenderowane w folderze **tmp**). Drugim ważnym elementem jest **File Format**. Standard to **PNG**, ale musimy to zmienić. Jeżeli zostawimy **PNG**, to każdą klatkę animacji wyrenderujemy jako




osobny obrazek. Będzie z tego bardzo dużo śmieci. Klikając w **File format**, najlepiej wybrać **AVI JPEG**. Dzięki czemu animacja zostanie zapisana do jednego filmu w formacie **AVI**.

7 Kliknijmy na **Render Image** lub naciśnijmy F12. Wyrenderujemy w ten sposób

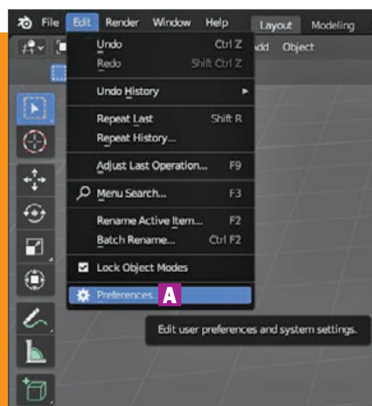
pojedynczy obrazek. Gdy pojawi się okno z wyrenderowanym obrazkiem, na górze zobaczymy informacje o renderze. W pozycji **Time** widzimy, ile czasu zajęło wyrenderowanie tej jednej klatki (obrazka). Im więcej obiektów i bardziej skomplikowanych shaderów, tym dłuższy czas renderowania. Można w prosty sposób policzyć, ile będzie mniej więcej trwało wyrenderowanie całej animacji. Wystarczy pomnożyć czas dla jednej klatki przez liczbę klatek. Nie jest to może idealne wyliczenie, ponieważ w trakcie trwania animacji położenie kamery i to, co ona aktualnie renderuje, może się zmienić, ale pozwala na podstawową estymację.

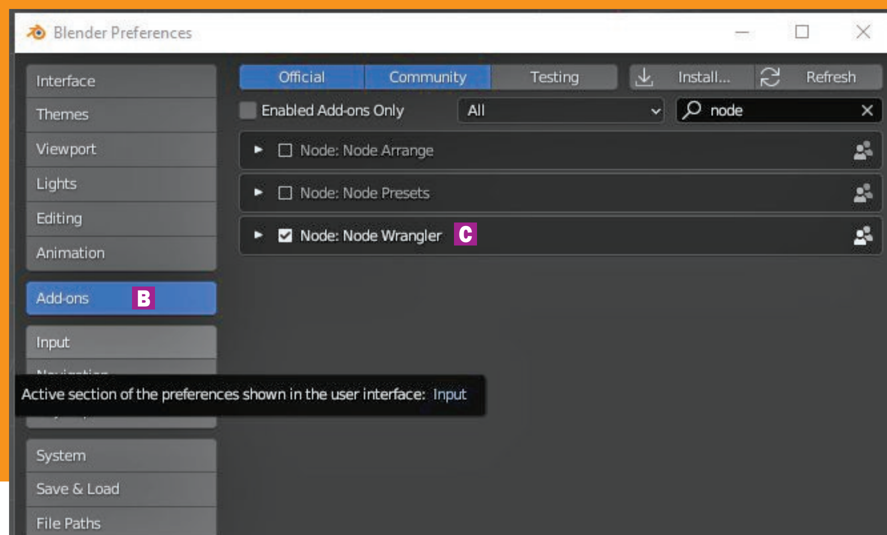
**Warto wiedzieć:** Na czas renderowania ma też wpływ rozdzielczość. Im wyższa, tym dłuższy czas renderowania.

**8** By wyrenderować naszą animację, klikamy na **Render**, **Render Animation** lub naciskamy ctrl+F12.

## PRZYDATNY DODATEK

W omówionym projekcie używaliśmy bloczków **Mapping** i **Texture Coordinate**. Blender pozwala na tworzenie i aktywowanie pewnych węzłów za pomocą skrótów klawiaturowych. By to zrobić, musimy zainstalować dodatek do Blendera. Klikamy na **Edit**, **Preferences** A i przechodzimy do części **Add-ons** B. Tutaj znajdziemy wiele dodatków do Blendera, interesuje nas **Node: Wrangler** C. Po aktywowaniu go będziemy mogli korzystać ze skrótów i ułatwień. Na przykład naszą powyższą kombinację dodamy teraz do bloczka za pomocą skrótu ctrl+T, a bloczek **mixRGB** dodamy za pomocą ctrl+0. Opis dodatku jest pod adresem: **docs.blender.org/manual/en/latest/addons/node/node_wrangler.html**



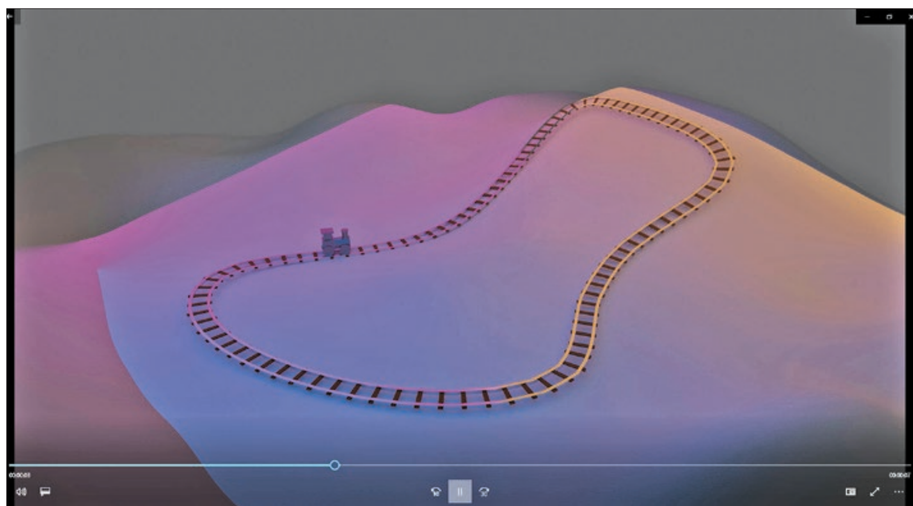
# 4 Własne animacje krok po kroku




**W tym rozdziale zobaczymy, jak korzystając z wiedzy zdobytej w poprzednich rozdziałach, stworzyć od podstaw własne animacje: lokomotywę jadącą po torach, falującą wodę i mrówkorobota**

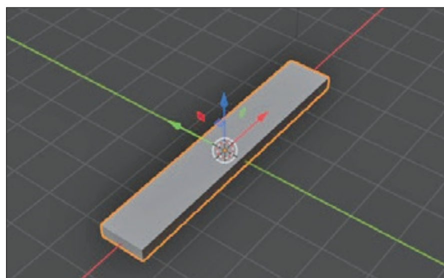
## Lokomotywa na szynach

By w praktyce poznać działanie Blendera, na początek przygotujemy łatwą animację na tak zwanych krzywych. Stworzymy szyny i prosty tor oraz lokomotywę, a potem wprawimy ją w ruch, tak by jechała wyznaczoną trasą.

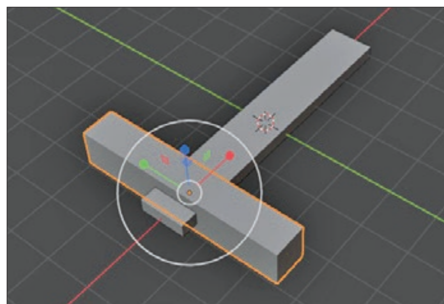


## Podkład i szyny, czyli tor dla kolejki

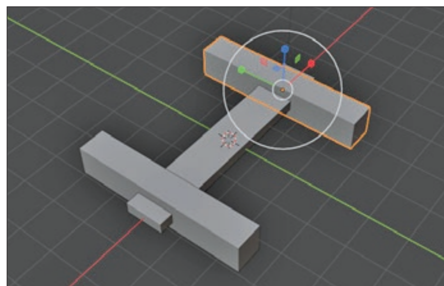
**1** Po uruchomieniu Blendera podstawowy Cube zmienimy w podkład dla szyn. Spłaszczamy go i wydłużamy. Ważne jest, byśmy wydłużali go względem czerwonej osi X. To ona odpowiada za kierunki prawo-lewo. Natomiast zielona oś Y jest osią przód-tył.



**2** Następnie dodajemy kolejny Cube. Jego również spłaszczymy i wydłużymy, tym razem jednak w osi Y. To będzie jedna z szyn.

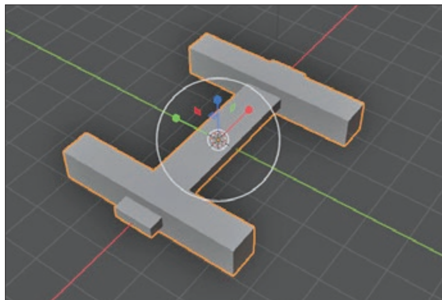


**3** Teraz trzeba zrobić drugą. Wystarczy, że zduplikujemy pierwszą szynę skrótem [shift]+[D], a następnie klikniemy prawym



przyciskiem myszy. By szyna była równo ustawiona, w oknie **Properties** wchodzimy do zakładki **Object Properties** (ta z pomarańczowym kwadratem). Znajdujemy tam właściwość **Location X**. Wystarczy teraz, że usuniemy minus, a nasza druga szyna powinna się ustawić po drugiej stronie podkładu.

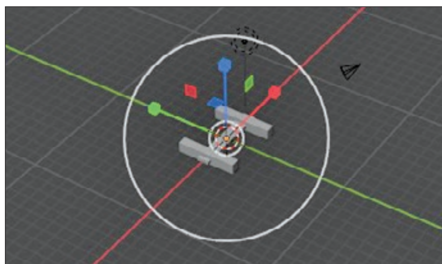
**4** Połączmy teraz te wszystkie elementy w jeden. Przytrzymujemy klawisz [shift] i zaznaczamy szyny i podkład. W Blenderze ważna jest kolejność zaznaczania, więc jako na ostatni klikamy na podkład. Chcemy, by to on wyznaczał środek modelu. Gdy mamy już wszystko zaznaczone, używamy skrótu klawiaturowego [ctrl]+[J]. Możemy w drzewku obiektów zmienić nazwę całego elementu na **Tor**.



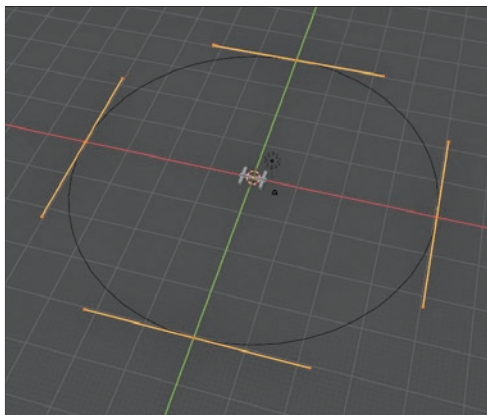
## Układamy trasę

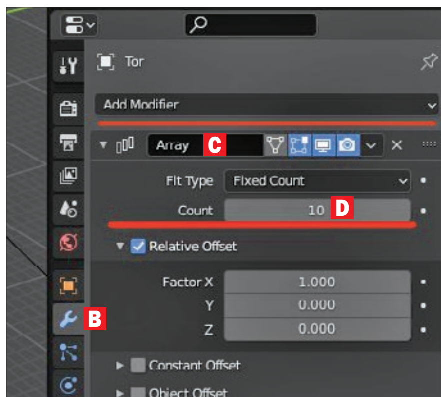
Mamy gotową podstawę. Teraz przygotujemy krzywą, na której będziemy rozkładać kopie stworzonego właśnie fragmentu toru, tworząc całą trasę.

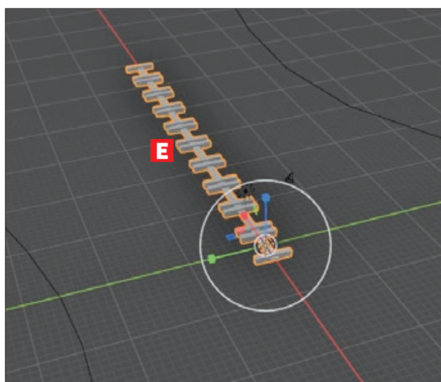
**1** Dodajemy nowy obiekt. Z okna wyboru wybieramy **Curve**, **Circle**. Pojawi się wtedy malutkie kółeczko. Powiększamy je.



**2** Przechodzimy teraz do zakładki **Modeling**. Zobaczymy, że na naszym kółku są cztery punkty.



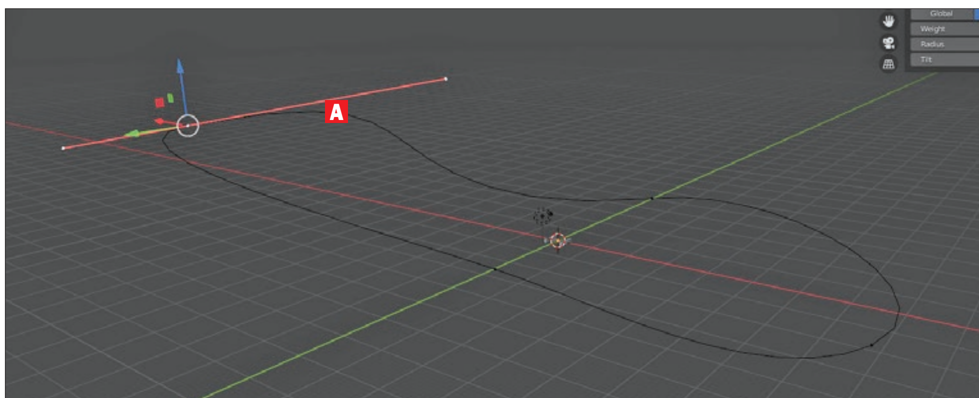


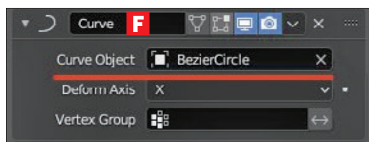




**3** Zaznaczając je, możemy modyfikować ich położenie, dzięki czemu zniekształcimy kółko **A**.

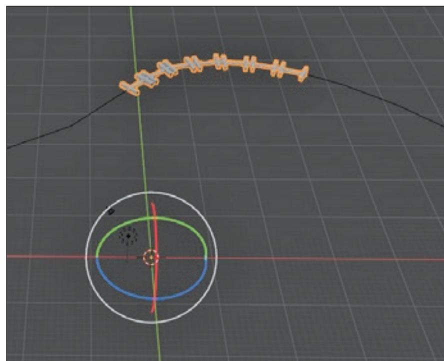
**4** Gdy utworzymy kształt, który nam odpowiada, wracamy do zakładki **Layout**. Zaznaczamy z powrotem nasz tor i w **Properties** przechodzimy do zakładki **Modifier Properties** (to ta z ikoną klucza płaskiego **B**). Tam z listy **Add Modifier** wybieramy modyfikator **Array** **C**. Modyfikator ten pozwala na wielokrotne powielanie obiektu w wybranym kierunku. Zwiększając wartość **Count** **D**, robimy więcej kopii **E**.

**5** Chcemy jednak, by tory układały się wzdłuż naszej krzywej. Musimy więc dodać kolejny modyfikator o nazwie **Curve** **F**. Gdy go dodamy, musimy wskazać, po jakiej krzywej ma się układać nasz obiekt. Klikamy

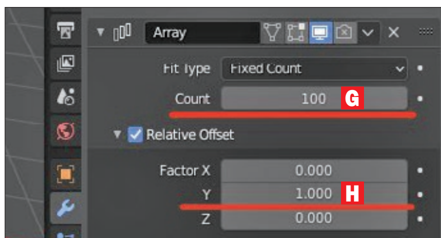





więc na pole **Curve Object** i z listy wybieramy naszą krzywą.



**6** Tor układa się bokiem, musimy go więc obrócić. Przechodzimy do zakładki **Object Properties** i tam ustawiamy wartość **Rotation Z** na **90**. Następnie wracamy do edycji modyfikatorów. W **Array** mamy właściwość **Relative Offset**. Musimy w nim zmienić wartość X na 0, a Y na 1. Dzięki temu nasze tory ustawią się wzdłuż linii. Ustawiamy następnie wartość **Count** G na taką, by zapełniła całą krzywą. Jeżeli tory w pewnym momencie zaczynają nachodzić na siebie, zmniejszamy trochę wartość Y w **Relative Offset** H.

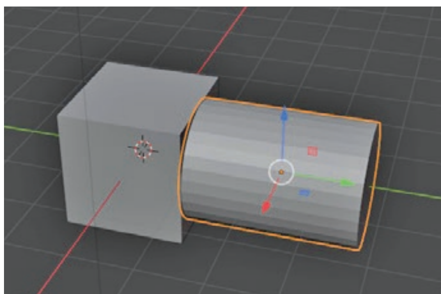


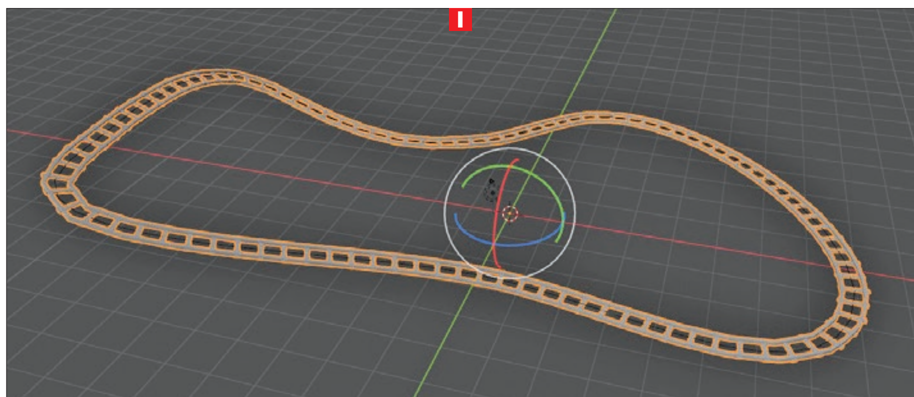

**7** Uzyskaliśmy właśnie tor I, po którym będzie się poruszać kolejka. Jeżeli teraz zmodyfikujemy naszą krzywą, tor powinien ustawić się według jej nowego położenia.

## Lokomotywa

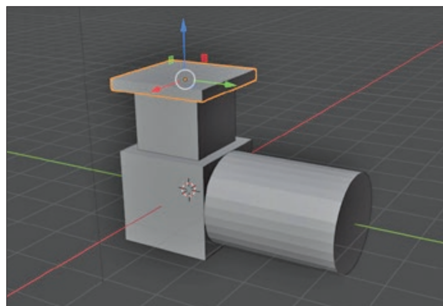
Stworzymy teraz prosty model lokomotywy. Nie będzie on dokładny, będzie raczej przypominał taki zbudowany z klocków.

**1** Dodajemy prosty **Cube**, a następnie **Cylinder**. Będą one tworzyły podstawę maszyny.

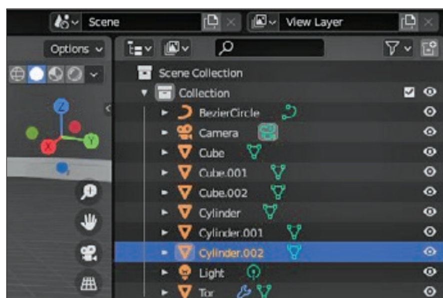


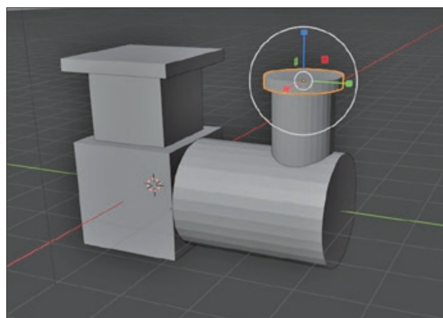


**2** Następnie tworzymy kolejny **Cube**, który podnosimy powyżej tego pierwszego i lekko pomniejszamy. To będzie kabina lokomotywy. Kolejny **Cube** podnosimy i spłaszczamy, tworząc daszek.

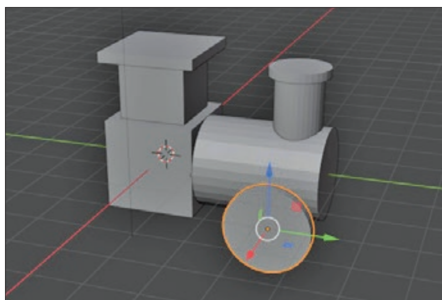


**3** Teraz komin. Zrobimy go podobnie jak kabinę, tylko zamiast Cube'ów użyjemy **Cylindrów**.

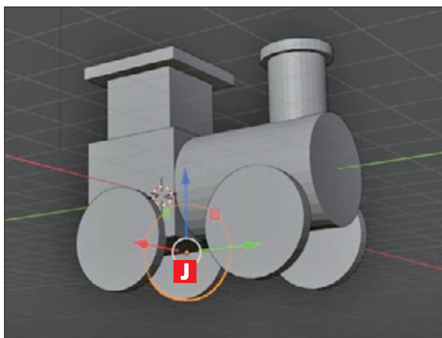


**4** Pozostaje nam zrobić koła. Również przygotujemy je z **Cylindrów**. Dodajemy jeden, spłaszczamy, obracamy i ustawiamy na boku naszej lokomotywy.

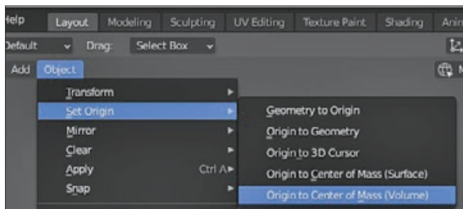


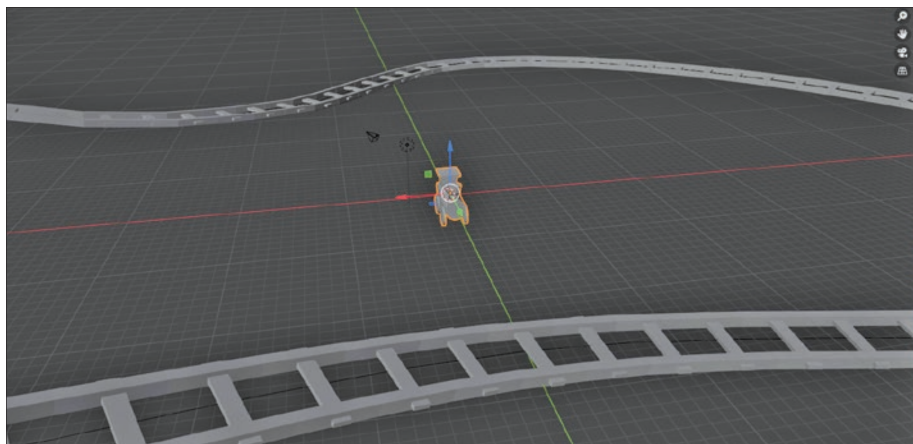
**5** Teraz wystarczy zduplikować stworzone koło i umieścić nowe koła z tyłu oraz po drugiej stronie lokomotywy.



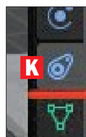

**6** Mamy gotową lokomotywę, jednak należy zwrócić uwagę, że wszystkie jej elementy są osobnymi obiektami. Musimy je połączyć tak, jak to zrobiliśmy poprzednio z torem. Zaznaczamy wszystkie po kolei, a następnie używamy skrótu klawiaturowego ctrl+J, by połączyć wszystkie elementy.

**7** Może się okazać, że środek ciężkości nie jest idealnie w środku. To ta pomarańczowa kropka **J**. By to poprawić, trzeba skorzystać z narzędzia **Object**, wybierając **Set Origin**, **Origin To Center Of Mass (Volume)**.



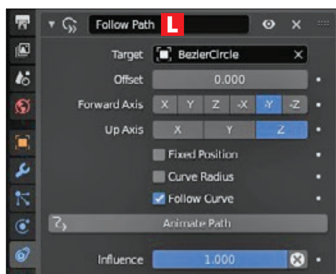


## Umieszczamy lokomotywę na torach i animujemy

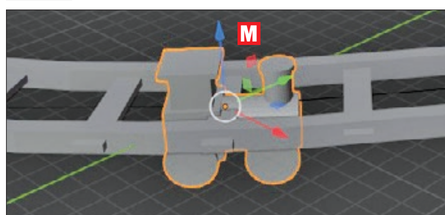



**1** Teraz pozostaje nam postawić lokomotywę na torach. Musimy przejść do zakładki **Object Constraint Properties** **K**.

**2** Zakładka ta wygląda podobnie do zakładki **Modifier**. Tutaj też dodamy właściwość z listy. Interesuje nas **Follow Path** **L**.

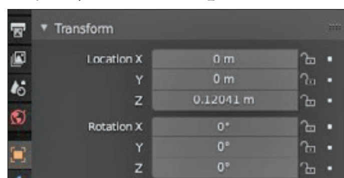



Ta właściwość pozwoli naszej lokomotywie poruszać się wzdłuż krzywej. W polu **Target** wybieramy naszą krzywą. Ustawi to

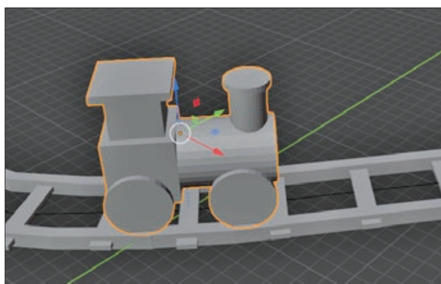



lokomotywę na krzywej. Jeżeli zdarzy się, że lokomotywa nie znajdzie się w torze jak na obrazku poniżej **M**, to trzeba zmienić jej położenie.

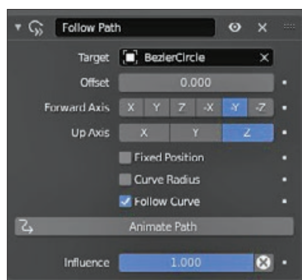
**3** W zakładce **Object Properties** musimy ustawić wartości **Location X**, **Y** i **Z** na **0**. Właściwość **Follow Path** oblicza położenie na krzywej względem punktu **0,0,0**. Każ-



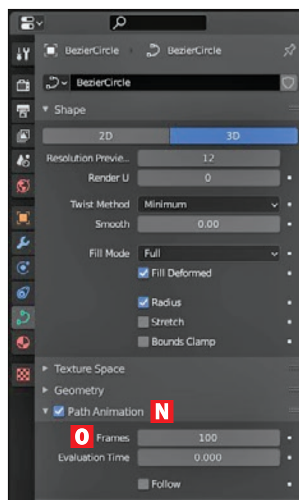

de przesunięcie wpływa na jego położenie względem naszej ścieżki. Gdy ciuchcia jest już ustawiona, możemy ją delikatnie podnieść, by znajdowała się na torze, oraz, jeżeli jest taka potrzeba, możemy ją powiększyć.



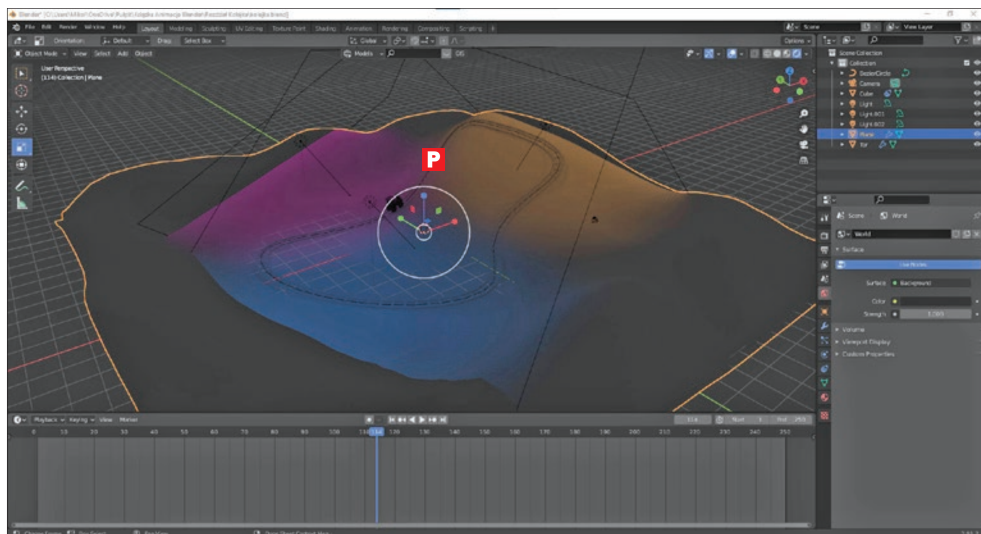
**4** Wracamy do **Follow Path**. Mamy tu właściwość **Forward Axis**, która będzie określać, gdzie jest przód. **Up Axis** analogicznie określa, gdzie jest góra. Ważne jest też pole **Follow Curve**. Gdy je zaznaczymy, nasza ciuchcia powinna się ustawić w odpowiednim kierunku, podążając za krzywą. Jeżeli wszystko jest gotowe, możemy sprawdzić całość, zmieniając wartość **Offset**. Jeśli wszystko wygląda dobrze, możemy przejść do animowania.




**5** Jednak zanim klikniemy na przycisk **Animate Path**, przejdźmy do naszej krzywej i otwórzmy zakładkę **Object Data Properties**. Znajdziemy tam zakładkę **Path Animation** **N**. Gdy ją rozwiniemy, zobaczymy pole **Frames** **O**. Określa ono, ile klatek zajmie przejście po całej krzywej. Im więcej klatek, tym dłużej ciuchcia będzie jechać.




**6** Gdy ustawimy odpowiadającą nam liczbę klatek, wracamy do lokomotywy i do **Constraints**. W naszym **Follow Path** klikamy na **Animate Path**. Możemy teraz zobaczyć całą naszą animację **P**. Jeżeli okaże się, że nasza lokomotywa jedzie bokiem albo tyłem, zmieniamy **Forward Axis**. Pozostaje nam już tylko wyrenderować animację. Możemy jeszcze dodać jakiś teren czy obiekt dookoła.



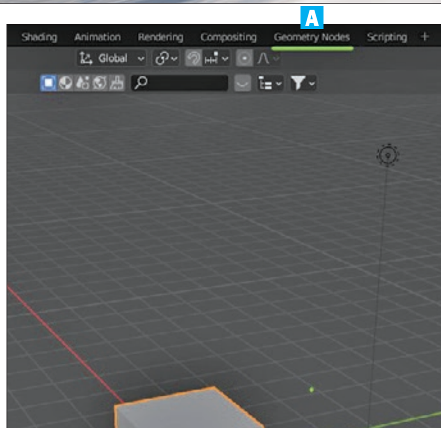
# Falowanie wody

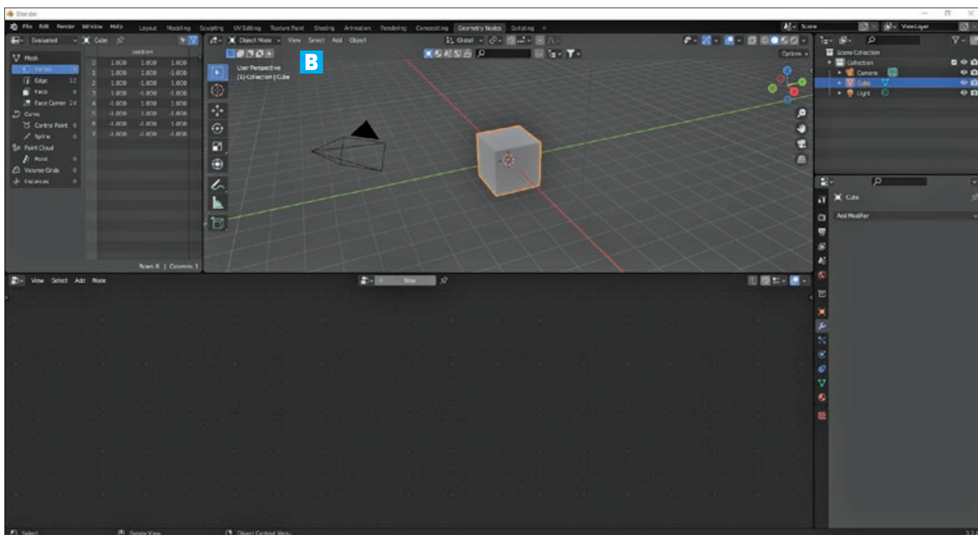


Ten projekt pokaże nam, jak korzystać z różnego rodzaju węzłów. Poznamy zarówno zakładkę **Shading**, jak i obecną od wersji 3.0 Blendera zakładkę **Geometry Node**. Stworzymy prostą symulację fal na płaszczyźnie.
**Uwaga!** Ta instrukcja przygotowywana jest w wersji Blendera 3.2.0. Inne wersje mogą mieć zmienione właściwości niektórych węzłów.

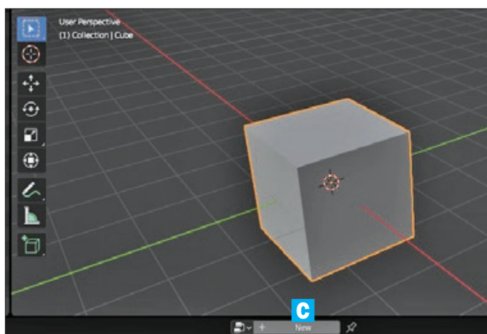
## Geometry Node – tworzymy nowy projekt

**1** Zacznijmy od stworzenia pustego projektu. Zaznaczamy nasz **Cube** i przechodzimy do zakładki **Geometry Nodes** A. Zobaczymy okno B. Jest to nowe narzędzie, wprowadzone od wersji 3.0 Blendera. Pozwala w sposób podobny jak bloczki z zakładki **Shading** zarządzać geometrią. Umożliwia generowanie kształtów i ich modyfikowanie.

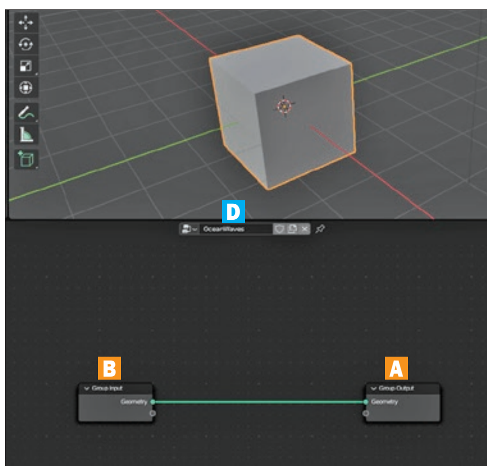


**2** Teraz musimy kliknąć na przycisk **New** **C**.

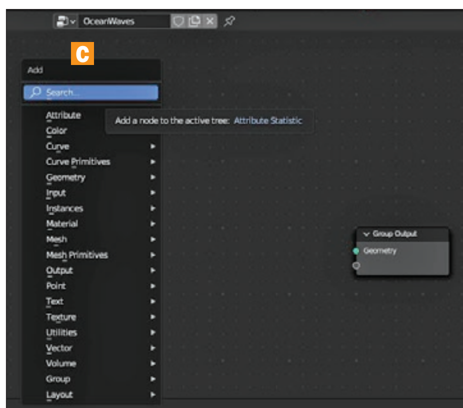



**3** Utworzy się nowy graf. W miejscu, gdzie znajdował się przycisk **New**, powstanie pole do nazwania naszych węzłów **D**. Nazwijmy je **OceanWaves**.

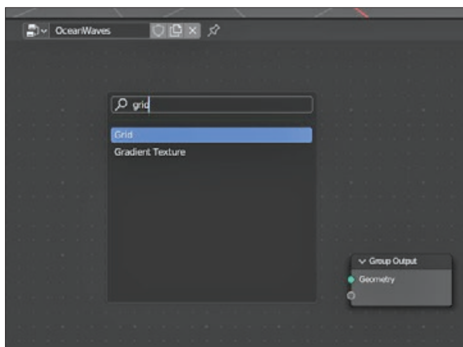



## Tworzymy siatkę

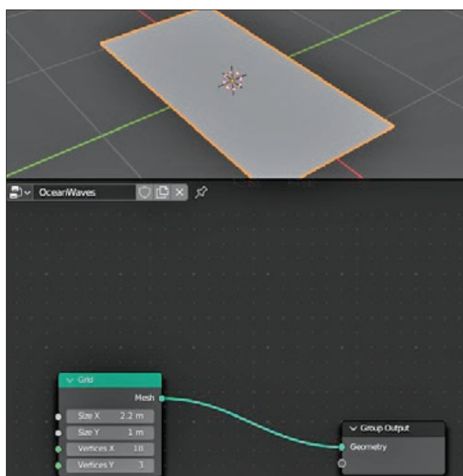
**1** Widzimy już dwa pierwsze węzły. **Group Output** **A** to blok, który jest odpowiedzialny za wyświetlenie końcowego wyniku. **Group Input** **B** odpowiada zaś za wprowadzane z zewnątrz dane, w tym obecny kształt modelu. Usuńmy więc Group Input. Zobaczymy, że model zniknął. Teraz możemy wybrać podstawowy model. Stworzymy siatkę. Używając skrótu klawiaturowego [shift]+[A], włączymy listę dodawania węzłów **C**.




**2** W górnej części widać **Search**. Skorzystamy z niego, by wyszukać blok **Grid**.




**3** Pojawi się nowy bloczek. Jego pole **Mesh** podłączymy do pola **Geometry**.

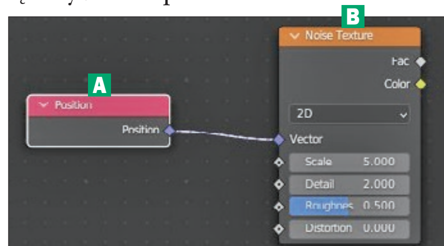


**4** Węzeł **Grid** ma cztery parametry. Dwa pierwsze to rozmiar w osiach X i Y, dwa ostatnie to liczba wierzchołków siatki względem tych osi. **Size** ustawmy na **5**, a **Vertices** na **100**.

## Tworzymy węzły

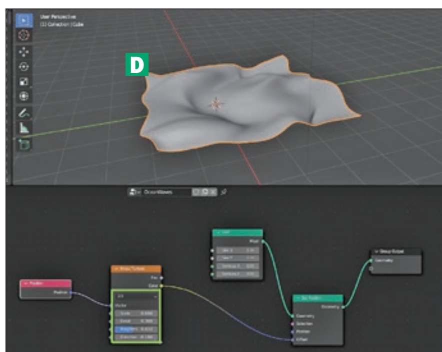
Przygotujemy teraz węzły, które trochę rozruszają naszą siatkę. Użyjemy bloczków, które będą przesuwały poszczególne wierzchołki w górę i dół.

**1** Dodajmy dwa pierwsze bloczki, **Position** **A** oraz **Noise Texture** **B**. W teksturze wybieramy z rozwijanej listy **2D**. Połączmy bloczki polami **Position** – **Vector**.




**2** Na razie nie widać nic ciekawego. Dodamy więc blok, który pomoże nam zwizualizować projekt. Nazywa się on **Set Position** **C**. Gdy go dodamy, to jego **Geometry** podłączamy do **Group Output**. Do niego podłączamy **Grid (Mesh – Geometry)**, a także **Noise Texture** do **Offset (Color – Offset)**.

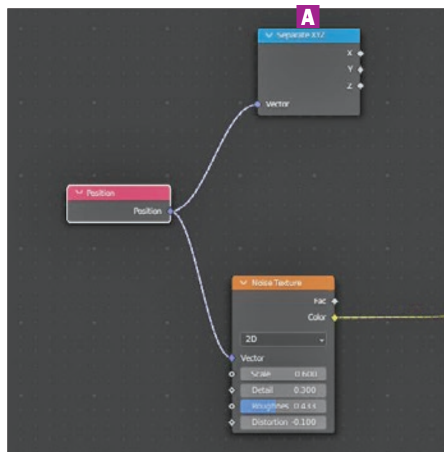
**3** Już widzimy pewne zmiany. Gdy zaczniemy modyfikować parametry w **Noise Texture**, będziemy mogli zobaczyć zmiany w wyglądzie naszej siatki **D**.



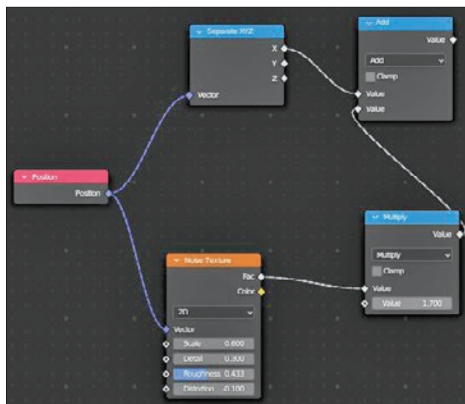
## Zmieniamy siatkę w fale

Żeby zrobić z siatki fale, będziemy musieli trochę pobawić się matematyką.

**1** Musimy dodać bloczek **Separate XYZ** **A**. To nam pozwoli za chwilę modyfikować ruch względem jednej z osi.

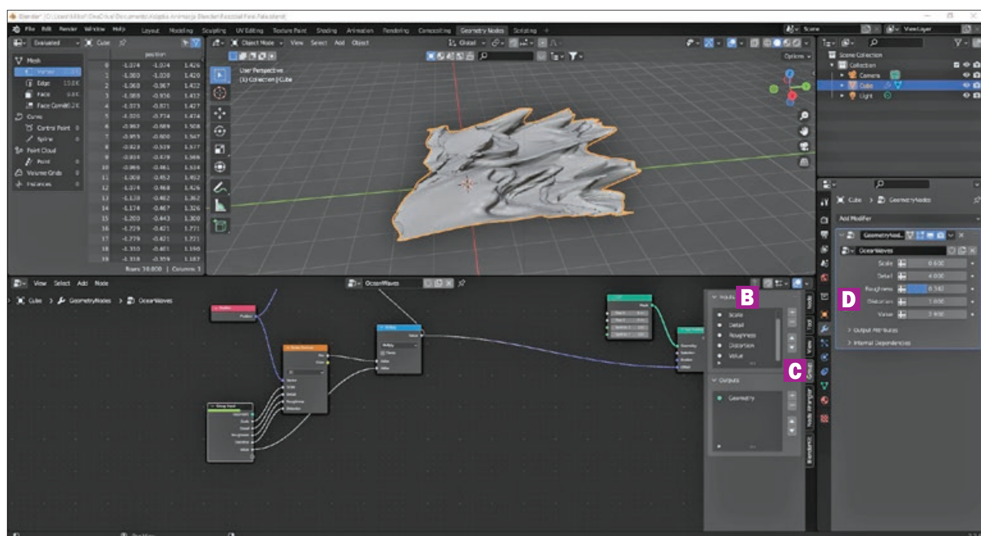


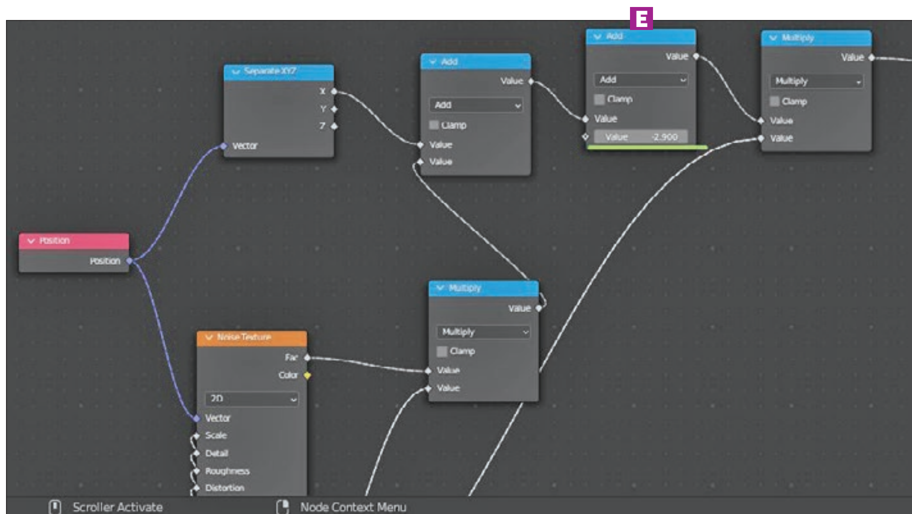

**2** Teraz posłużymy się bloczkiem **Math**. Dodajmy dwa takie bloczki, jeden ustawiamy z listy na **Add**, drugi na **Multiply**. Do **Multiply** dołączamy nasz **Noise Texture (Fac – Value)**, a potem **Separate XYZ**, z jego wyjścia **X**, oraz **Multiply** podłączamy do dwóch **Value** w **Add**. Następnie też nasze **Multiply** podłączmy do **Offset** w **Set Position**.




**3** Pomanipulujmy teraz suwakami. Zobaczmy, jak się zmienia nasz model.

**4** W przyszłości będziemy chcieli móc zmieniać właściwości niektórych pól. Dodamy więc bloczek **Group Input**. Gdy połączymy jego czarne kółko z jakąś wartością, wyświetli się pole do ustawienia w odpowiednim oknie. Naciskając klawisz



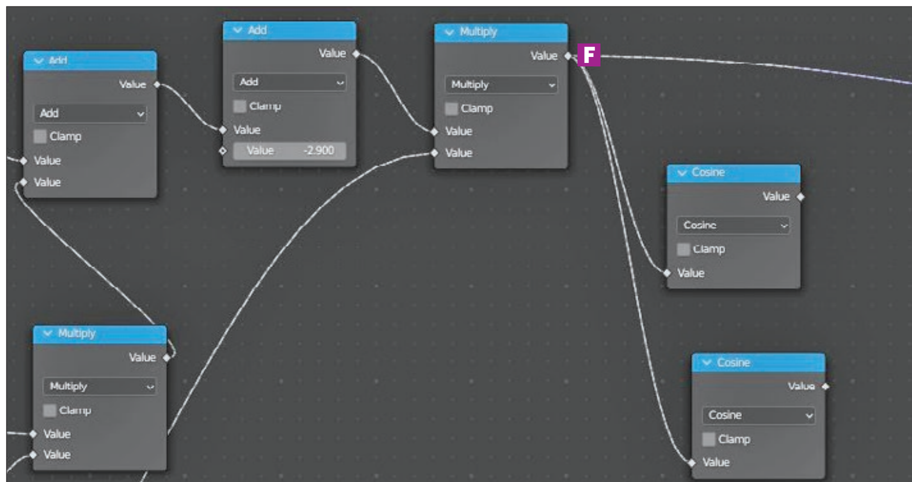


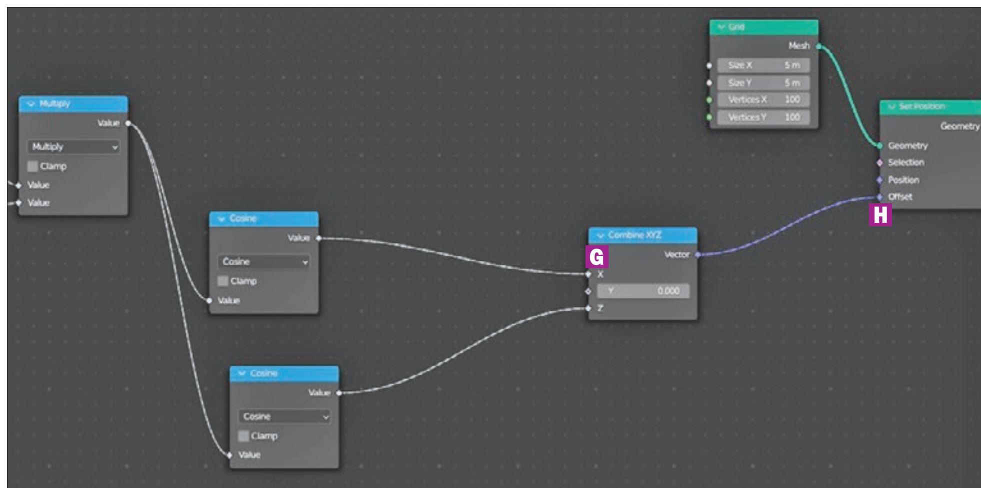

N, włączymy boczny panel **B**, gdzie w zakładce **Group** **C** będziemy mieli dostęp do nazw. Klikając dwa razy na wartość **Value**, będziemy mogli zmienić jej nazwę. Nazwijmy ją **Smoothness**. Natomiast w oknie modyfikatorów **D** zobaczymy modyfikator **Geometry Nodes** z naszymi węzłami i właśnie podłączonymi zmiennymi. Kolejne też będą się tutaj pojawiać.

**5** Dodajemy teraz kolejny węzeł **Add** i do niego podłączamy poprzednie **Add** **E**. Teraz dodajemy węzeł **Multiply** i ostatnie **Add** podłączamy do pierwszego **Value**, a do drugiego – pole z **Group Input**.

**6** Zaznaczone powyżej pole w bloczku **Add** posłuży w przyszłości do przesuwania fal. Dodajmy teraz dwie funkcje cosinus. Trzeba je także dodać poprzez bloczek **Math** i wybrać z listy opcję **Cosine**. Podłączamy do obydwu wynik z naszego ostatniego bloczka **Multiply** **F**.

**7** Teraz będziemy potrzebowali bloczka, który przez wyliczone przez nasze bloki

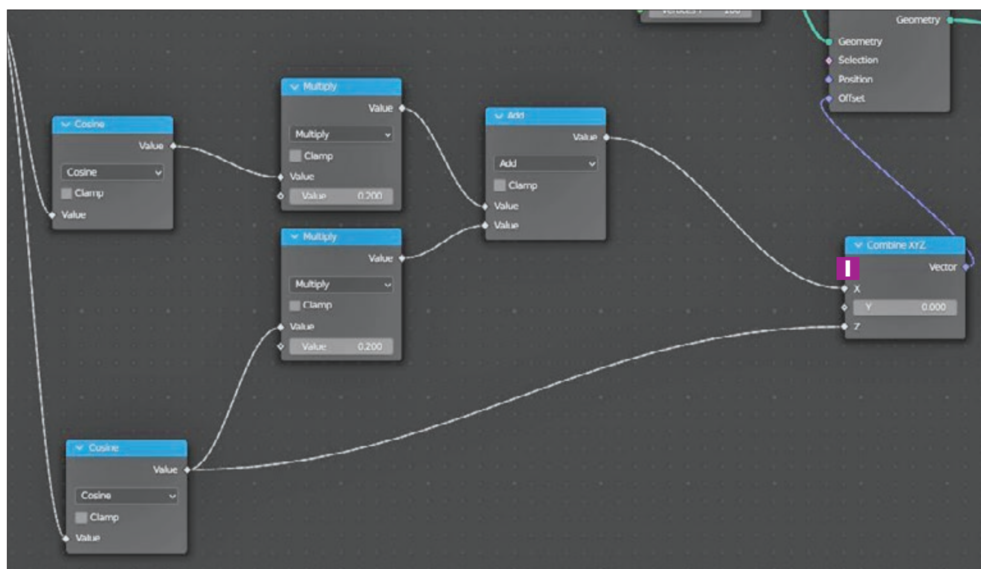


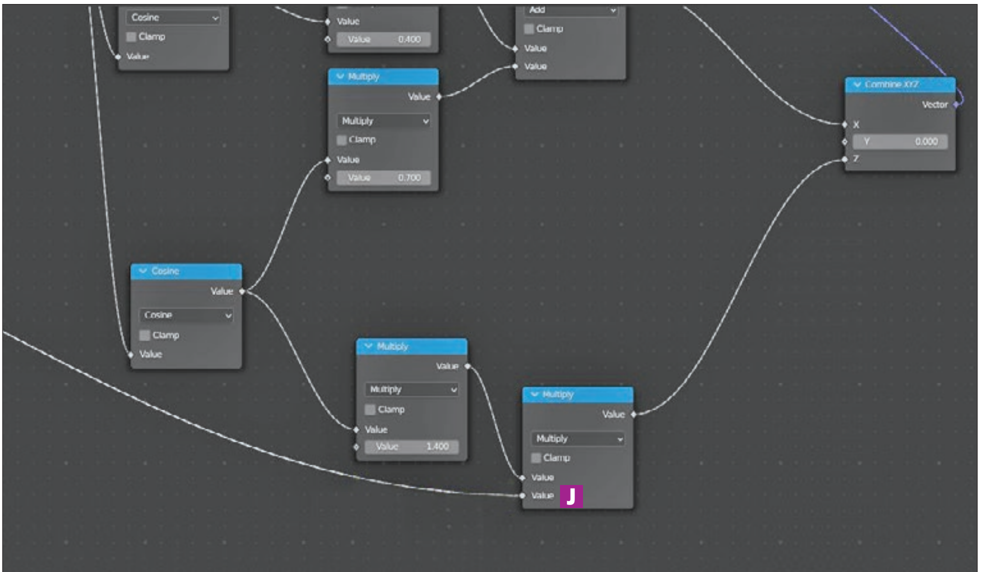



wartości będzie przesuwał falę przez siatkę. Posłużymy się węzłem **Combine XYZ**. Cosinusy podłączamy kolejno do **X** i **Z** **G**, a **Combine** do **Offset** **H** w **Set Position**.

**8** Gdy zmieniamy parametry, nasza siatka już o wiele bardziej przypomina rzeczywistą falę. Ustawmy przesunięcia dla X. Dodajmy dwa **Multiply** i do każdego podepnijmy jeden z cosinusów. Następnie oba te **Multiply** dodajmy węzłem **Add**, a wynik podłączmy do **Combine XYZ** **I**.

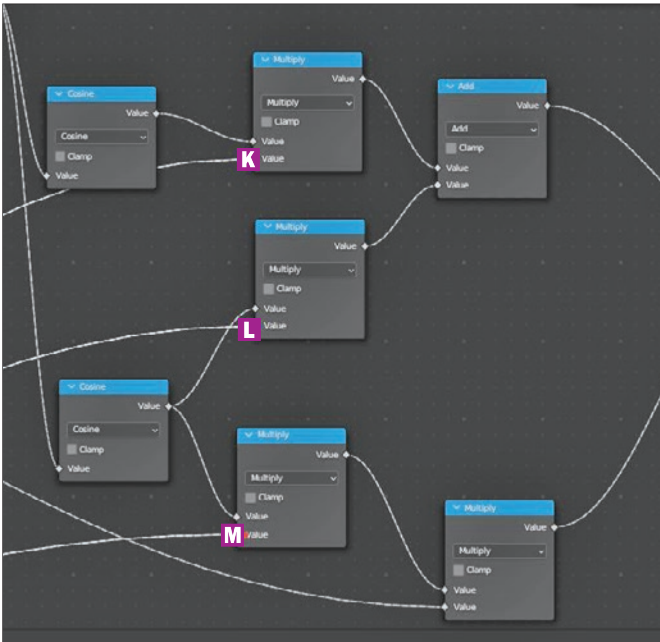
**9** Teraz dodamy zarządzanie wysokością. Dodajemy dwa nowe **Multiply** i najpierw z cosinusa dolnego wartość przypisujemy do pierwszego, a jego wynik przypisujemy do drugiego **Multiply**. Z niego wyprowadzimy



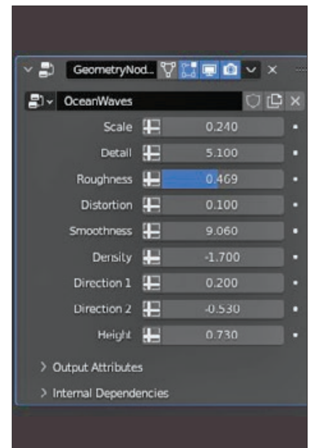


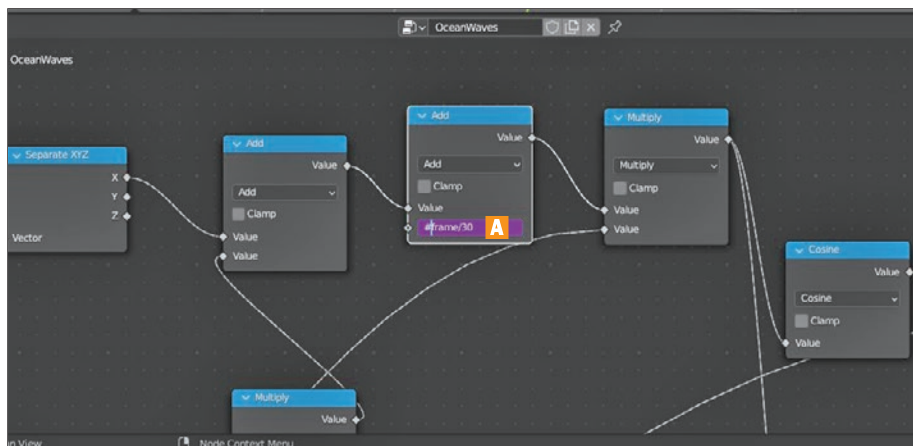

wartość do **Z** w **Combine XYZ**. By uszczegółowić animację, do ostatniego **Multiplay** podłączamy również **Fac** z **Noise Texture** **J**.

**10** Pozostaje nam jeszcze dodać trzy pozostałe wolne wartości w bloczkach **Multiply** do naszego **Group Input**. Nazwiemy je odpowiednio Direction 1 **K**, Direction 2 **L**, Height **M**.




**11** Ustawmy parametry tak jak poniżej.






## Animowanie

Pozostaje nam tylko wprawić fale w ruch.

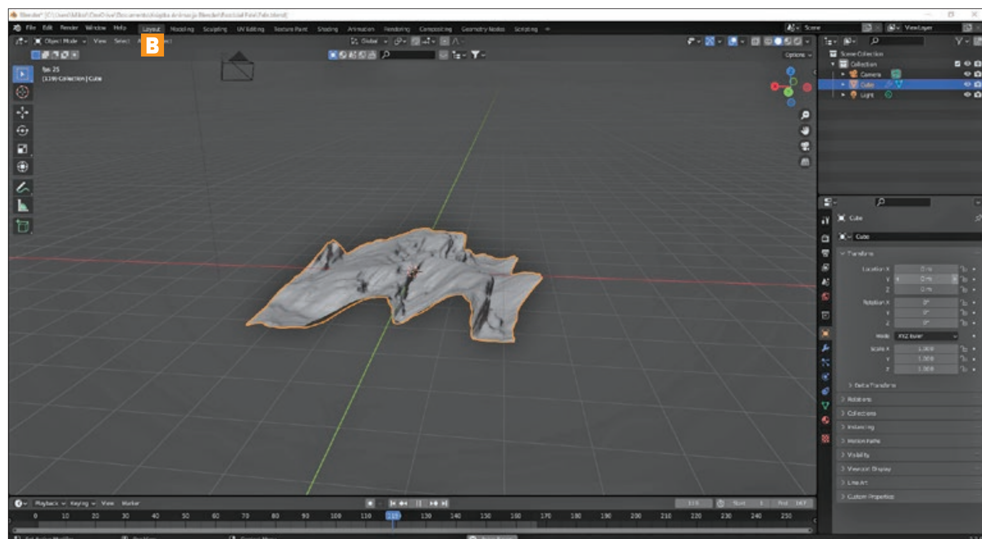
**1** W polu **Add**, które jako jedyne pozostało bez podpięcia, klikamy na wartość **Value** i wpisujemy instrukcję: **frame/30** **A**. Liczba może być inna – im niższa, tym fale szybciej się poruszają.

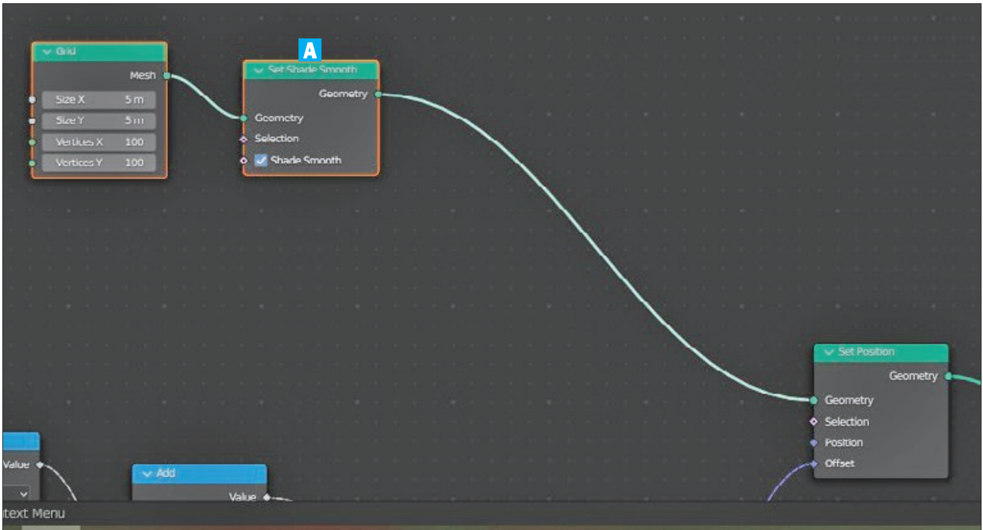
**2** Przenosimy się teraz do **Layout** **B** i uruchamiamy animację. Nasze fale poruszają się wraz z animacją. Można zwiększyć liczbę klatek animacji, żeby wydłużyć jej czas trwania.

## Wygładzanie i materiał

Nasz model jest jednak kanciasty. Poligony nie są wygładzone, więc nie otrzymujemy naturalnego efektu. Trzeba to dopracować.

**1** Kliknięcie prawym przyciskiem myszy i wybranie **Shade Smooth** nie daje rezultatu. Musimy wrócić do **Geometry Nodes** i dodać bloczek o nazwie **Set Shade Smooth**

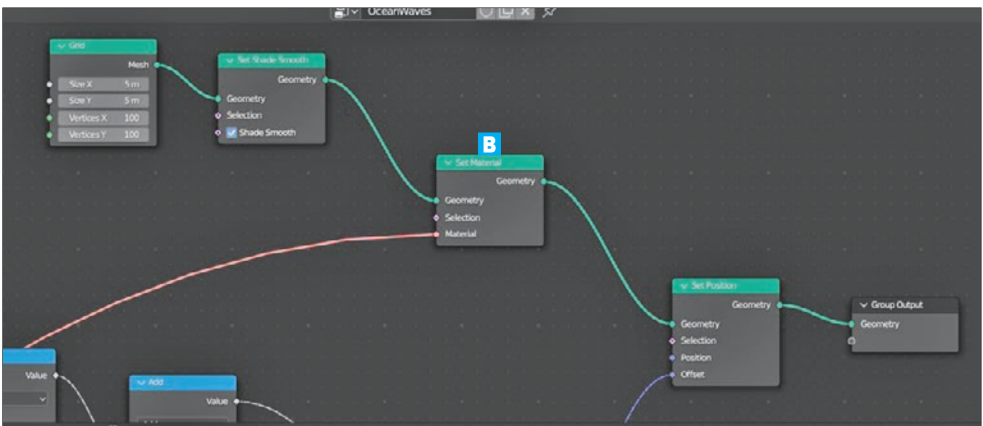


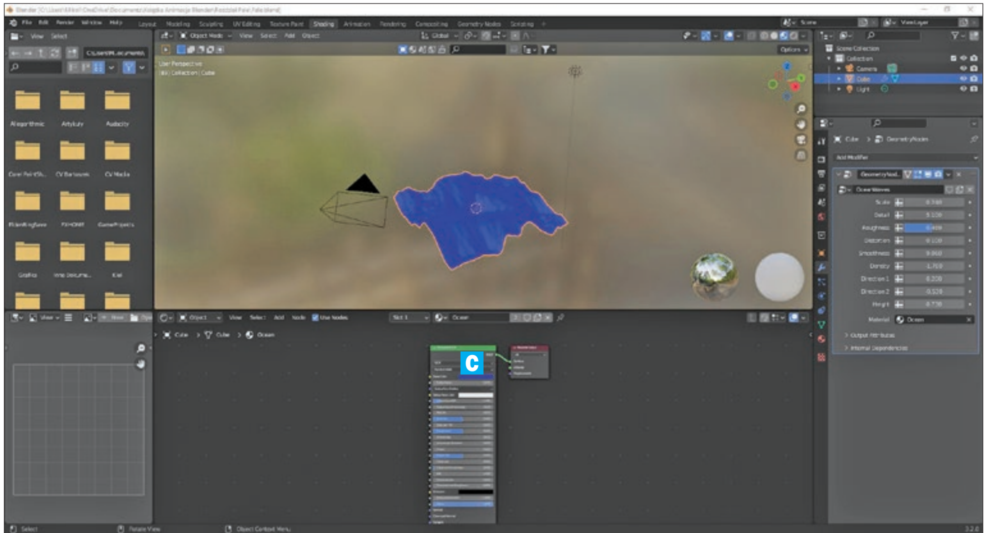



**A**. Dlaczego tak się dzieje? Otóż stworzona przez nas siatka nie jest takim modelem jak zwyczajnie dodany **Cube** czy **Plane**. Dlatego należy dodać wygładzenie poprzez osobny węzeł między **Grid** a **Set Position**.

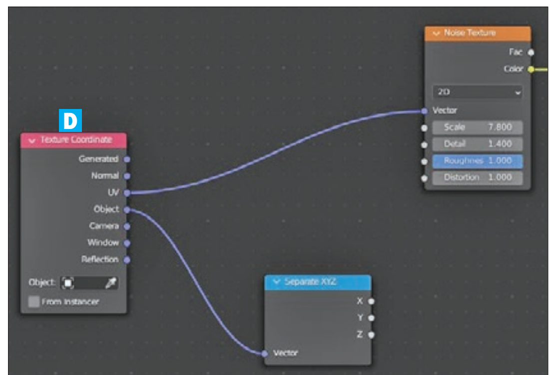
**2** Niestety, ta sama zasada dotyczy również materiałów nakładanych na model. Dopóki nie dodamy węzła, który będzie takie materiały przechowywał, nie będziemy mogli ich dodać. Więc najlepiej od razu dołożyć węzeł **Set Material** **B**. Układamy go pomiędzy **Set Shade Smooth** a **Set Position**. Od razu też możemy podłączyć jego pole **Material** do **Input Group**.

**3** Przejdźmy do zakładki **Shading**. Wyglądem i sposobem pracy przypomina **Geometry Nodes**. Klikamy na **New** i tworzymy nowy materiał. Nazwiemy go **Ocean**. Gdy utworzymy ten materiał, od razu podepnijmy go w polu **Material** w naszym modyfikatorze **Geometry Nodes**. W polu **Base Color** możemy wtedy wybrać na próbę dowolną barwę **C** (kolejna strona), by zobaczyć, czy materiał rzeczywiście został podpięty.

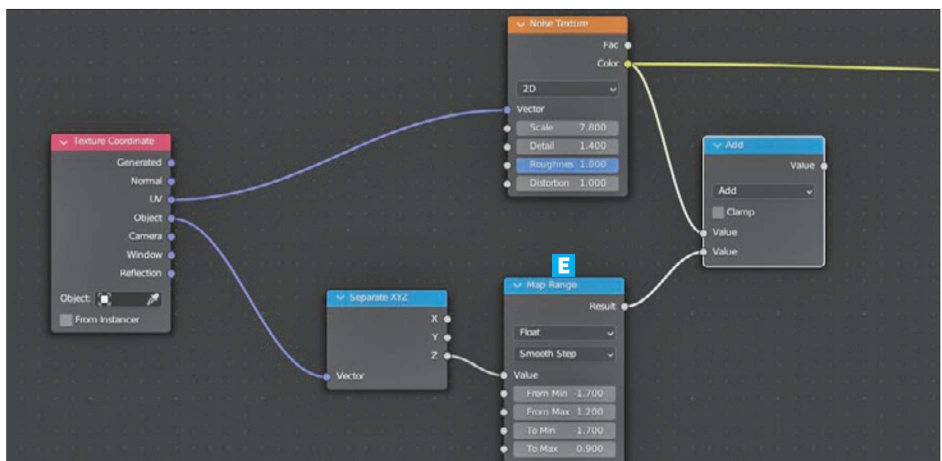


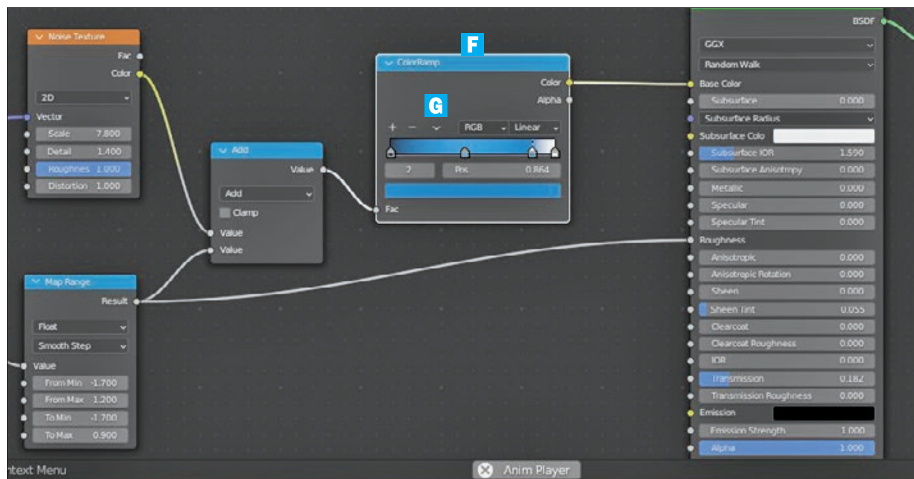



**4** Teraz będziemy potrzebowali bloczka o nazwie **Texture Coordinate** D. Dodamy także dwa bloki, **Noise Texture** i **Separate XYZ**. **UV** połączymy z **Noise Texture**, gdzie ustawimy z listy **2D**. **Separate** połączymy z **Object**.




**5** Nie daje to na razie żadnego rezultatu. Dodajemy kolejny węzeł **Map Range** E. Ten bloczek pozwala konwertować wartość. Wartości **From** to ograniczenia zakresu, który wchodzi do naszego węzła. Każda większa

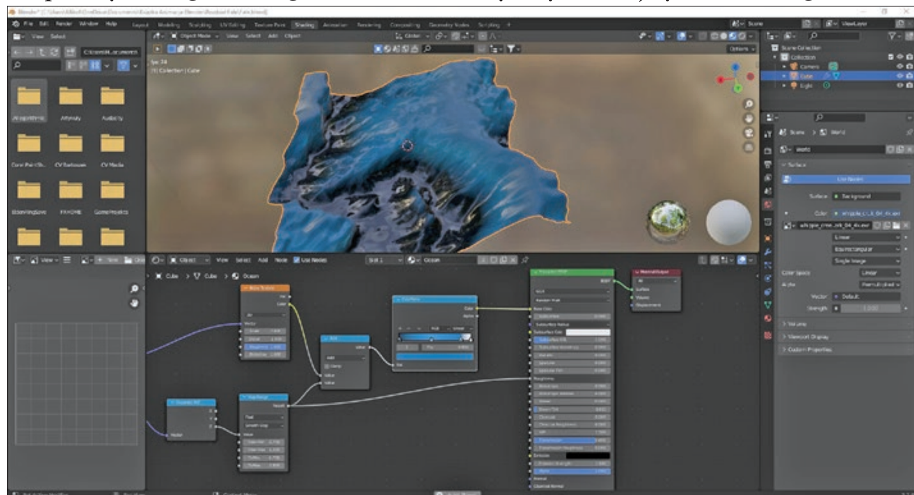





jest ustawiana na **Max**, a mniejsza na **Min**. Następnie wszystkie te wyniki są skalowane do wartości **To**. Zatem jeśli wcześniej najwyższy punkt przy **From [-1,1]** to było jeden, to przy **To [0,4]** wszystko jest przesunięte tak, by było równo rozłożone i najwyższy punkt to cztery. Teraz oba nasze węzły łączymy za pomocą **Add**. Tak samo jak przy **Geometry Nodes**, tu również dodajemy go za pomocą **Math**.

**6** Pozostaje nam ostatni węzeł. Jest nim **ColorRamp** **F**. Służy on do mapowania wartości kolorami z podanego gradientu. Podpinamy do niego z naszego **Add** wartość do **Fac**, a następnie dodajemy kolory za pomocą **G**, po lewej stronie ciemne, po prawej jasne. Teraz możemy podpiąć **Color** z **ColorRamp** do **Base Color**. Połączmy też **Map Range** do **Roughness**.

**7** Gotowe, pobawmy się teraz wartościami, by uzyskać jakiś ciekawy efekt. Poniższe ustawienia przygotowane są tak, by niższe warstwy były ciemniejsze, natomiast na samym szczycie ma się stworzyć piana, która powinna być biała. Jeżeli chcemy zrobić na przykład iluzję płynnego metalu, możemy wartość **Metallic** ustawić na 1. Pokombinujmy i wyrenderujmy coś ciekawego.



# Kości



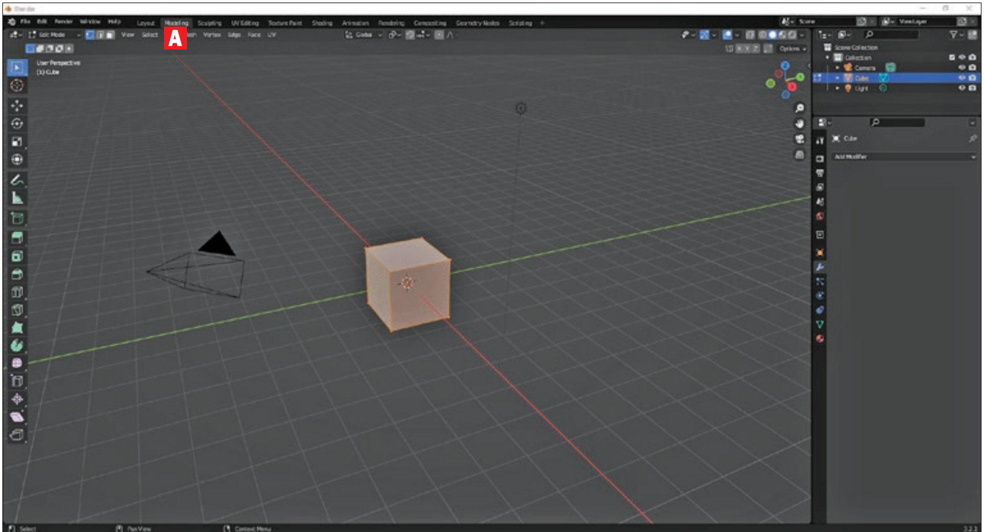
Kiedy animujemy człowieka, a także inne żywe, poruszające się istoty, musimy zadbać o to, by postać poruszała się naturalnie. Na przykład kolana powinny zginać się w odpowiednim miejscu i to tak, by to zgięcie wyglądało jak zgięcie ciała, a nie, jakby noga złamała się nagle w połowie i miała zaraz wrócić do stanu poprzedniego.
Stworzono system, który pozwala na przygotowanie kontrolerów, które imitują na przykład szkielet. Nawiązując do anatomii, nazywamy te elementy **kośćmi**. Całość możemy potem przypisać do modelu. Proces dodawania kości do modelu nazywamy **rigowaniem**. Co ciekawe, animację szkieletową możemy też przygotować w programach do grafiki 2D, nie tylko w 3D.
Dużo gier jest dziś tworzonych z wykorzystaniem kości. Ta metoda pozwala na stworzenie bazy animacji i wygenerowanie na jej podstawie wszystkich potrzebnych stanów, jak skok czy kucanie. Przy zwykłej animacji poklatkowej trzeba każdy z nich rysować osobno wraz z przejściami.

# Nasz własny ogonek, czyli poznajemy kości

Zanim zaczniemy większy projekt, zrobimy ćwiczenie, które pozwoli oswoić się z kośćmi. Stworzymy ogonek.

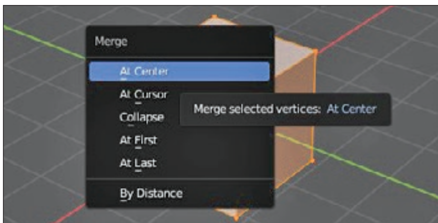
**1** Uruchamiamy nowy projekt w Blenderze. Klikamy na sześcian, a następnie przechodzimy do zakładki **Modeling** **A**.

**2** Teraz jesteśmy w edycji siatki. Potrzebujemy pojedynczego wierzchołka. By go uzyskać, połączymy wszystkie w jeden. Muszą być wszystkie zaznaczone (podświetlone na pomarańczowo). Jeżeli przez przypadek się „odkliknęły”, wystarczy, że użyjemy klawisza [A] na klawiaturze. Pozwala on na

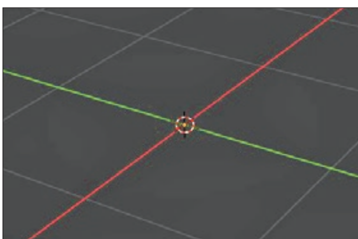



zaznaczenie wszystkiego lub usunięcie zaznaczenia. Aby połączyć zaznaczone wierzchołki, naciskamy klawisz M na klawiaturze.

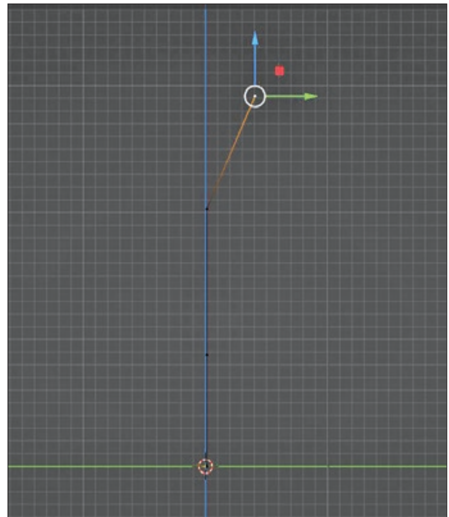
**3** Pojawi się lista, z której wybieramy **At Center**. Pozwoli to na połączenie wszystkich wierzchołków w jeden, w wyliczonym środku między nimi.



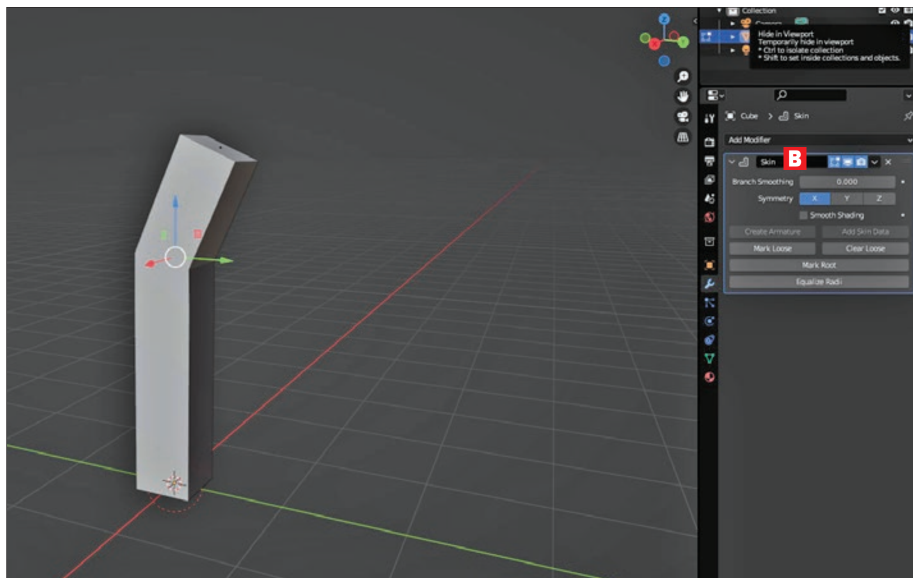

**4** Mamy teraz tylko jeden wierzchołek. Narysujemy linię. Klikamy na wierzchołek i naciskamy klawisz E na klawiaturze.



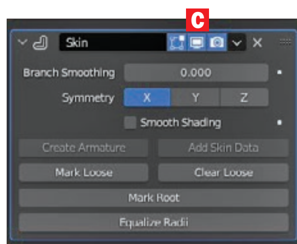
Spowoduje to włączenie narzędzia **Extrude**, które pozwala na wyciąganie zaznaczenia. Zróbmy tak trzy razy. Naciskamy E, wyciągamy, odklikujemy i powtarzamy czynności. Uzyskamy krzywą linię.



**5** Kiedy już ją mamy, dodajemy modyfikator **Skin** **B**. Pozwoli on nam na wymodelowanie na bazie naszych wierzchołków potrzebnej nam bryły.

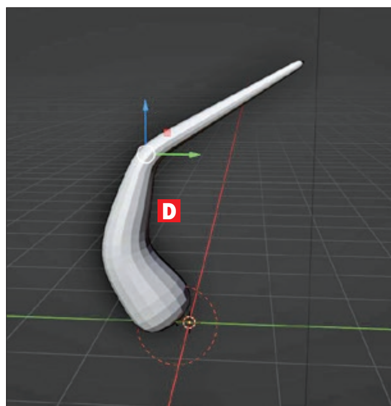



**6** Gdy zaznaczymy któryś z wierzchołków i będziemy go przesuwać, to nasz model będzie się również przesuwał. Jeżeli nie widzimy wierzchołków, można na chwilę wyłączyć podgląd modyfikatora. Wystarczy kliknąć na przycisk z monitorem **C** na panelu.

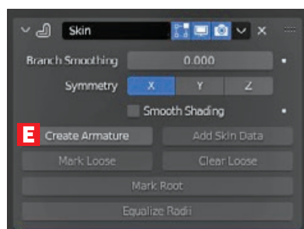



**7** Jeżeli dodamy modyfikator **Subdivision Surface**, to nasz model będzie bardziej przypominał walec. Możemy też zwęzić lub poszerzyć poszczególne elementy. Tutaj nie działa zwykłe skalowanie, ponieważ wierzchołek nie ma skali. Możemy jednak dostosować jego promień, używając skrótu ctrl+A. Tak powstanie nasz ogonek **D**.

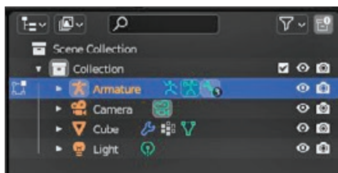
**8** Możemy wrócić do zakładki **Layout**. Będziemy mogli teraz użyć narzędzia



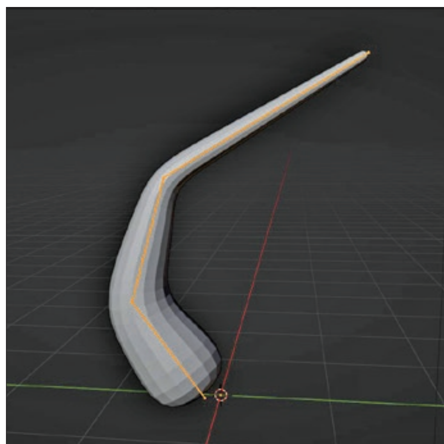

**Create Armature** **E**. Modyfikator **Skin** pozwoli nam automatycznie stworzyć kości dla naszego modelu na bazie wierzchołków.



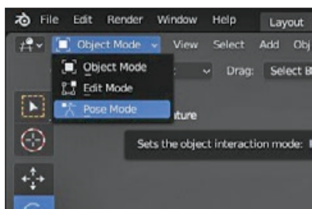
**9** Po użyciu tego narzędzia zobaczymy, że w outlinerze pojawił się nowy element o nazwie **Armature**.



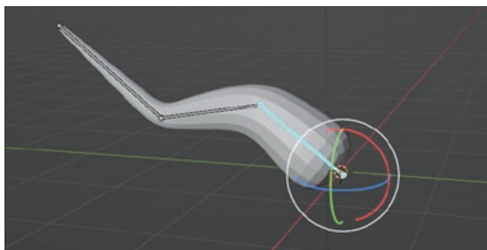

**10** Jest to szkielet utworzony na bazie wierzchołków. Na modelu będziemy mogli je podejrzeć jako osobną linię.



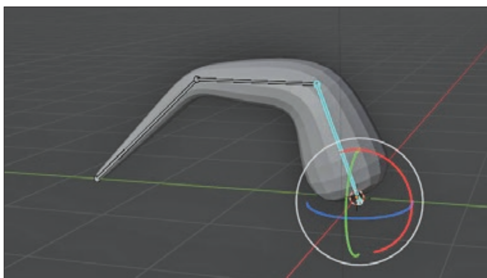
**11** Nie możemy jednak nim poruszać. Kiedy to zrobimy, szkielet zostawia za sobą model. W trybie **Object Mode** najlepiej zaznaczać oba elementy. Można też połączyć je w taki sposób, że jeden staje się dzieckiem drugiego (ale często i tak trzeba zaznaczyć oba). Jednak nam chodzi o poruszanie poszczególnymi kośćmi. Dlatego przechodzimy do trybu **Pose Mode**. Musimy go wybrać z listy w lewym górnym rogu sceny.



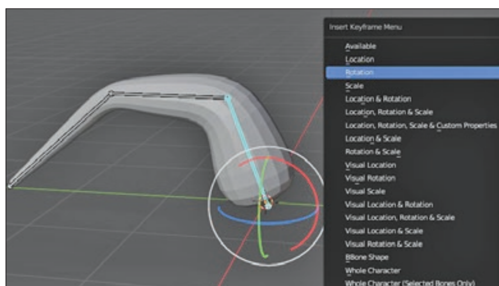

**12** Gdy do niego przejdziemy, będziemy mogli przeklikiwać się między kośćmi. Możemy również dokonywać modyfikacji kości. W dużej mierze, gdy animujemy kości, to tylko je obracamy. Czasami będziemy je przesuwać lub skalować, ale często będzie to burzyło naturalny stan szkieletu i podpiętego do niego modelu albo da po prostu efekt obrotu.

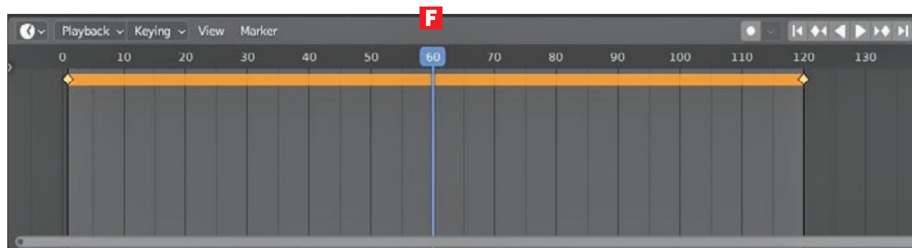


**13** Aby przygotować animację, ustawiamy ogonek w wybranej pozycji. Obracamy odpowiednią kość, aż uzyskamy interesujący nas efekt.

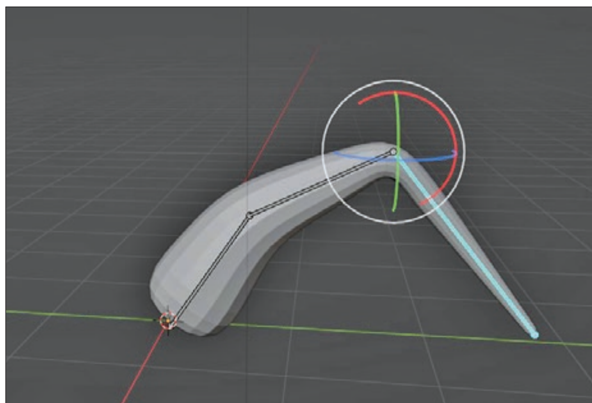


**14** Teraz zaznaczamy po kolei każdą kość i na osi czasu na klatce pierwszej naciskamy I na klawiaturze. Dodajemy klucz dla obrotu.



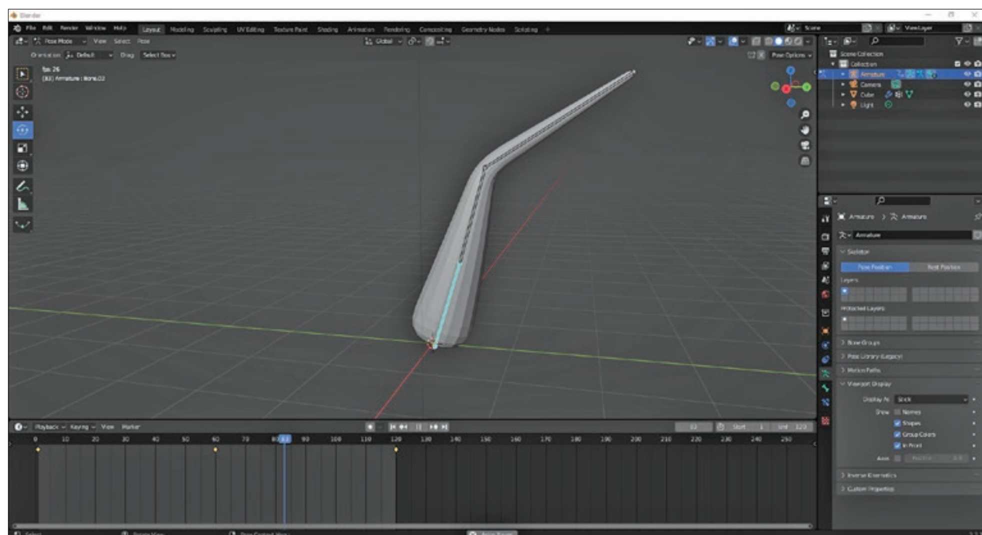



**15** Przechodzimy na ostatnią klatkę i robimy to samo. Dzięki temu nasza animacja będzie się zaczynać i kończyć w tym samym momencie.



**16** Pojawi się linia, która pokazuje, że w tej części osi czasu klucze nie zmieniają wartości. Zburzymy to jednak. Igłę na osi ustawiamy na środkowej klatce **F**. Pozycję ogonka ustawiamy na drugą stronę.

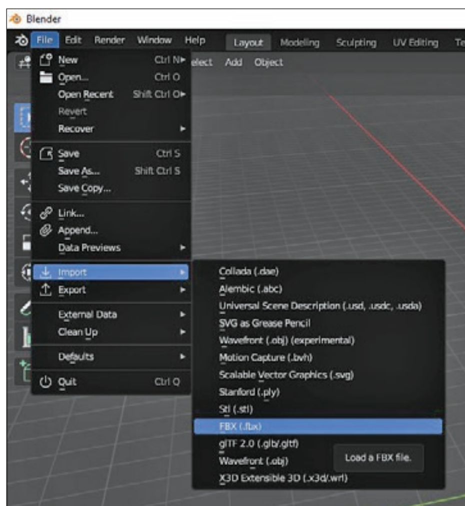
**17** I znów dodajemy klucze do każdej kości dla rotacji. Gdy już to zrobimy, możemy odtworzyć animację, by zobaczyć, co nam wyszło. Widzimy, jak nasz ogonek deformuje się pod wpływem kości. Brawo, oto pierwsza animacja na kościach. Jednak na tym nie poprzestaniemy. Teraz czas na coś zaawansowanego.



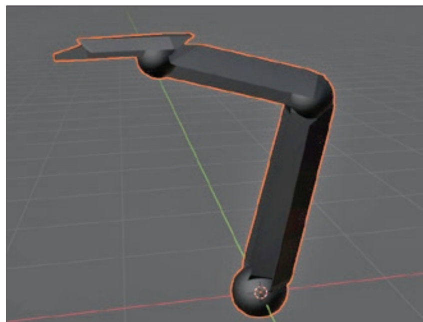
# Mrówkorobot (AntMachine)

## Import nogi robota

**1** Stwórzmy nowy projekt i usuńmy z niego cube, kamerę i światło. Na dołączonej do książki płycie są pliki w formacie **FBX**. Jest to najbardziej popularny format, do którego eksportuje się modele i animacje 3D. Potrzebujemy na początek modelu o nazwie **Leg**. By zaimportować do Blendera model, musimy kliknąć na **File**, **Import**, **FBX (.fbx)**.

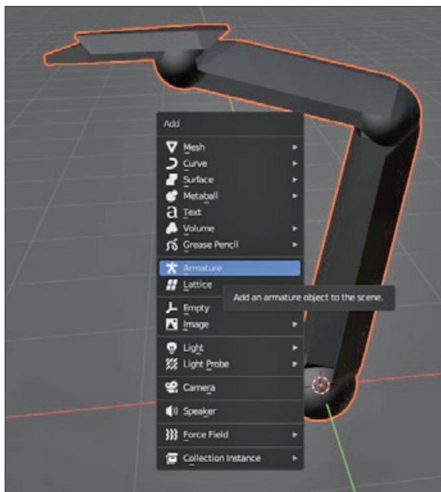



**2** Otworzy się okno, w którym trzeba znaleźć plik. Gdy go znajdziemy, zaznaczamy go i wybieramy **Import FBX**. Na naszej scenie pojawi się przygotowany wcześniej model nogi robota.



## Dodajemy kości

**1** Musimy przygotować kości do tej nogi. W przeciwieństwie do poprzedniego ćwiczenia, tutaj dodamy kości ręcznie. Naciskamy Shift+A i klikamy na **Armature**.

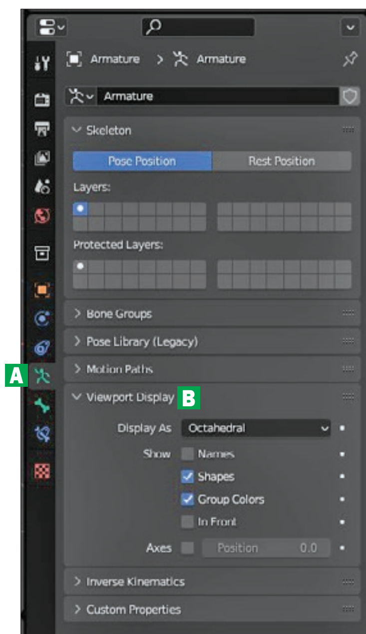



**2** Pojawi się pojedyncza kość. Wygląda zupełnie inaczej niż w poprzedniej wskazówce. Kształt będziemy mogli potem wybrać. Teraz pojawia się jednak mały kłopot. Otóż kość jest zasłaniana przez nogę. Kości nie są widoczne w renderze. Jednak gdy je edytujemy, to musimy wziąć pod uwagę, że ich położenie będzie wpływać na model.



Chcemy mieć je na czas edycji widoczne, na przodzie.

**3** Przechodzimy do zakładki **Object Data Properties** (w przypadku szkieletu ma ikonę patyczaka **A**). Następnie rozwijamy panel **Viewport Display** **B**.

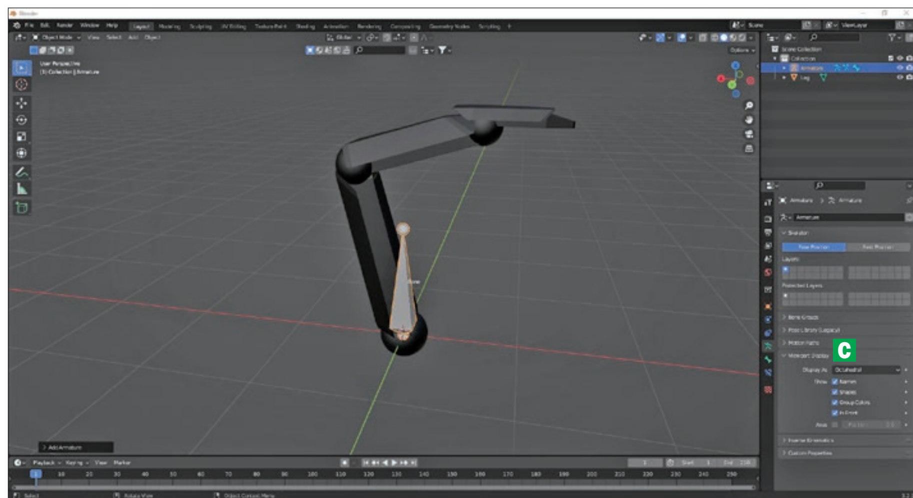


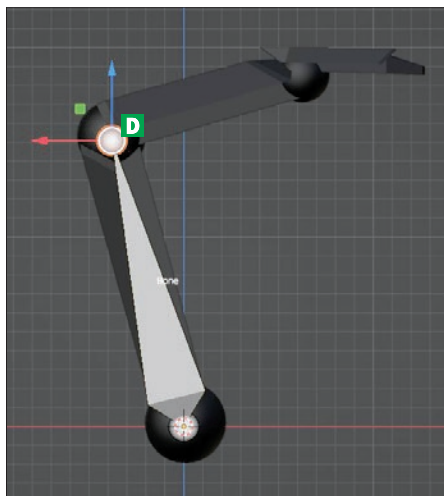

**4** Tam mamy kilka właściwości, które w tej chwili nas interesują **C**:

- **Display As** to sposób wyświetlania kości. Standardowo włączone jest **Octahedral**, co daje nam ten dziwny kształt. Do wyboru mamy też **Stick**, **B-Bone**, **Envelope** i **Wire**. Najlepiej wybrać taki, z którym będzie się nam najłatwiej pracowało.
- **Show In Front** pozwala na ustawienie widoczności kości na pierwszym planie. Dzięki temu to kość będzie zasłaniać model, a nie na odwrót.
- **Show Names** może nie przyda nam się tak bardzo, ale warto to wiedzieć. Pozwala na włączenie nazw wszystkich kości w szkielecie. Dzięki temu łatwiej na przykład przypisywać wybrane wierzchołki do wybranej kości.

**5** Przechodzimy do zakładki **Modeling**. Będziemy dodawać i układać kolejne kości. Na początek zajmiemy się pierwszą kością. Ustawiamy się bokiem, by móc rozmieszczać kości zgodnie z nogą. Klikając na górną kulkę **D**, możemy za pomocą przesunięcia wydłużyć kość. Przesuniemy ją również na kolano nogi.

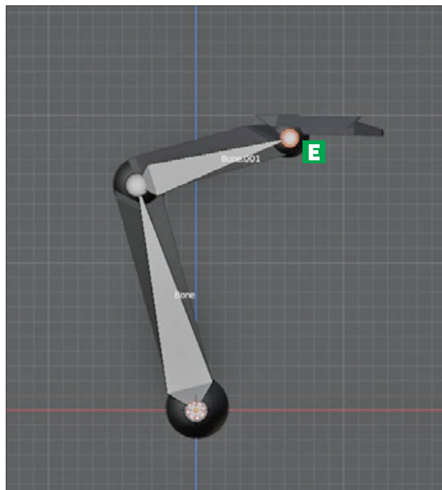
**6** Mamy już pierwszą kość. Teraz czas na następną. Dalej, mając zaznaczoną gór-





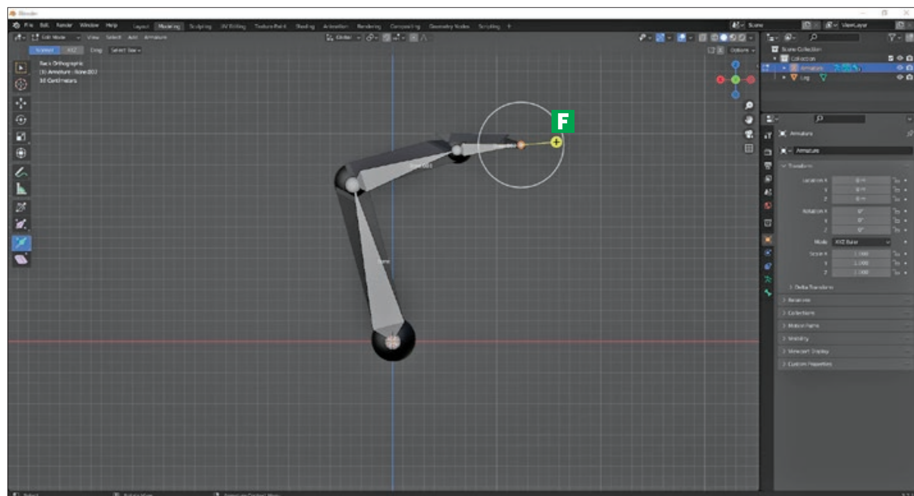

ną kulkę, użyjemy narzędzia **Extrude**. Naciskamy na klawiaturze klawisz E. Stworzy nam to nową kość, którą przeciągniemy do następnego złączenia w stopie **E**.

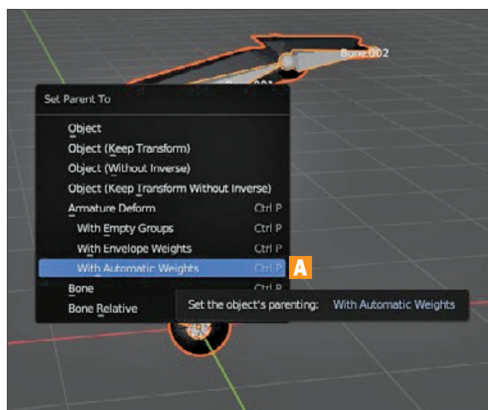
**7** Narzędzie **Extrude** możemy też włączyć, wybierając je w przyborniku po lewej. Gdy je wybierzemy, pojawi się manipulator. Łapiąc za żółtą część **F**, będziemy mogli wyciągnąć nową kość względem linii, którą on wyznacza. Jest jeszcze biały okrąg. Gdy za niego złapiemy, będziemy mogli wyciągnąć kość w dowolnym wybranym przez nas kierunku. Jeśli coś teraz nie będzie nam się zgadzało, mamy chwilę, by to poprawić.



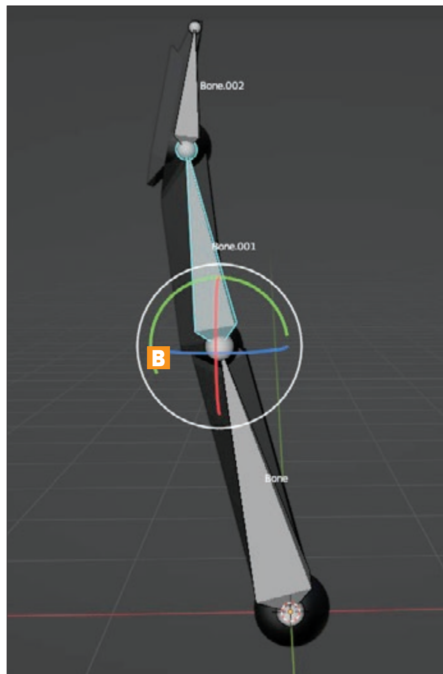

## Łączymy szkielet z modelem

**1** Wracamy do zakładki **Layout**. Teraz należy połączyć szkielet z modelem, czyli go zriggować. Musimy zaznaczyć najpierw nogę, a następnie z wciśniętym klawiszem ctrl zaznaczyć szkielet. Kolejność ma tu znaczenie. Pojawia się lista, z której wybieramy **Armature Deform**, **With Automatic Weights** **A**.





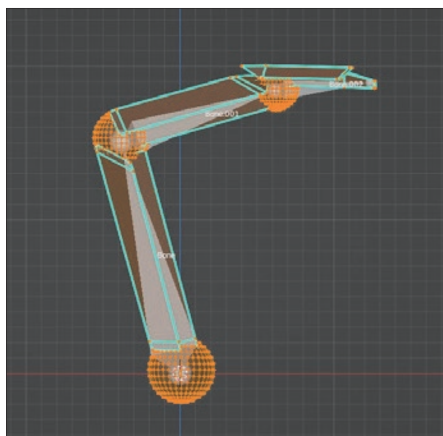

**2** Jak dokładnie działają w tym przypadku kości? Skąd wiedzą, którą część modelu mają poruszyć? Otóż wierzchołki modelu są przypisywane do wybranej kości i po prostu wierzchołek wie, która kość na niego oddziałuje. Zdarza się jednak, że na wierzchołek muszą działać dwie albo więcej kości – w różnym stopniu. Mówimy wtedy o wagach.




Wierzchołek jest przypisywany do wybranej kości z pewną wagą określającą, jak bardzo ruch danej kości na niego wpływa.

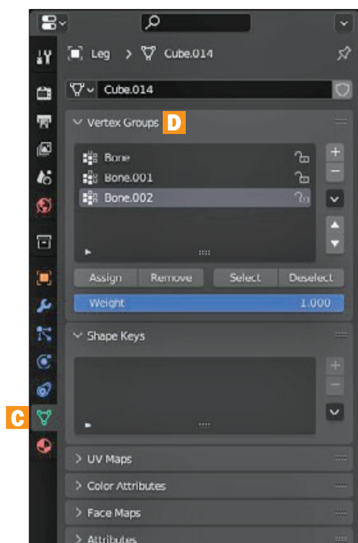
**3** Użycie **Automatic Weights** spowodowało, że Blender samodzielnie przypisał odpowiednie wagi do wybranych elementów modelu. Może to jednak skutkować efektami ubocznymi. Gdy przejdziemy do **Pose Mode** i będziemy poruszać kością, to, owszem, noga się poruszy, jednak spowoduje to też deformację części nogi **B**.

**4** Nasz model to noga robota, która nie powinna w taki sposób zmieniać kształtu. Wprowadzimy więc pewne poprawki. Zaznaczamy nogę, bez kości, i przechodzimy do zakładki **Modeling**.

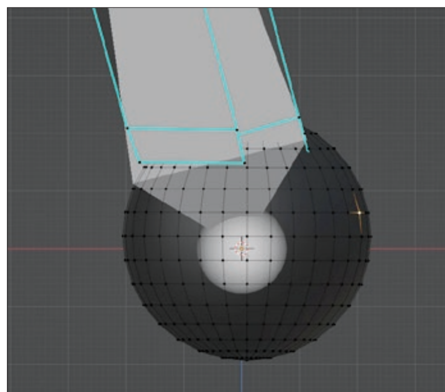


**5** Noga składa się z kilku połączonych z sobą siatek. Są jednym modelem, ale każdą z nich można osobno zaznaczyć i dzięki temu każdą z nich przypisać do wybranej kości. Przechodzimy do zakładki **Object Data Properties** **C** i odnajdujemy **Vertex Groups** **D**. Jest to narzędzie, które pozwala na grupowanie wybranych wierzchołków, by potem móc nimi manipulować.

**6** Powstały trzy grupy reprezentujące trzy kości. Musimy teraz po kolei zaznaczać wybraną siatkę i usuwać jej wierz-

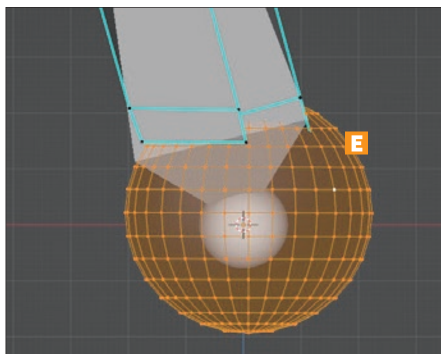



chołki z wybranych kości i przypisywać je do odpowiedniej. Zaznaczamy jeden wierzchołek na pierwszej siatce.

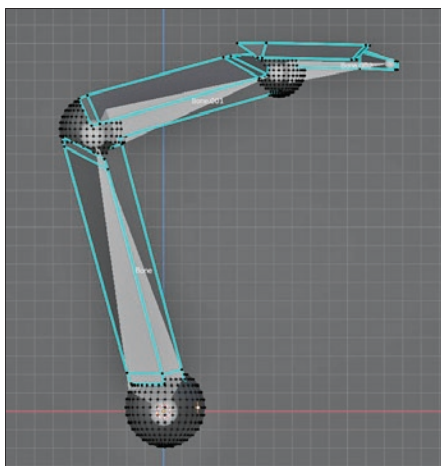


**7** Naciskamy klawisz L na klawiaturze. Dzięki temu zaznaczymy wszystkie wierzchołki należące do wybranej siatki E.

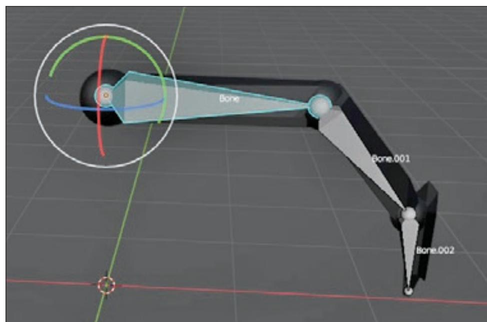
**8** Gdy mamy to gotowe, w panelu **Vertex Group** usuwamy te wierzchołki z kości **Bone.001** i **Bone.002**. Zaznaczamy je po kolei i naciskamy **Remove**. Następnie wybieramy **Bone**, czyli kość, do której te wierzchołki mają być przypisane, i klikamy na **Assign**.




Robimy tak z wszystkimi elementami nogi, przypisując je do odpowiednich kości.



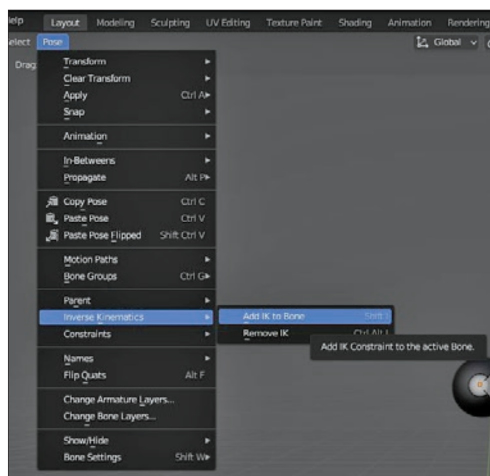
**9** Kiedy mamy to gotowe, możemy wrócić do zakładki **Layout** i w **Pose Mode** zobaczyć, jak wszystko działa.



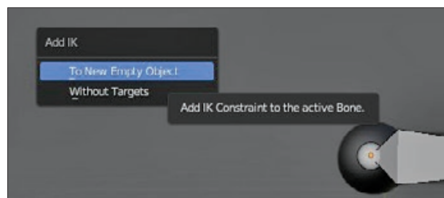
## Sterowanie nogą

Możliwości sterowania są trochę ograniczone. Musimy za każdym razem przechodzić do **Pose Mode** i cały ruch trzeba edytować klatka po klatce i kość po kości. Możemy jednak stworzyć narzędzie, które nie dość, że ułatwi nam animowanie, to jeszcze sprawi, że będziemy mogli przesuwać kończynę bez potrzeby wracania do **Pose Mode**.

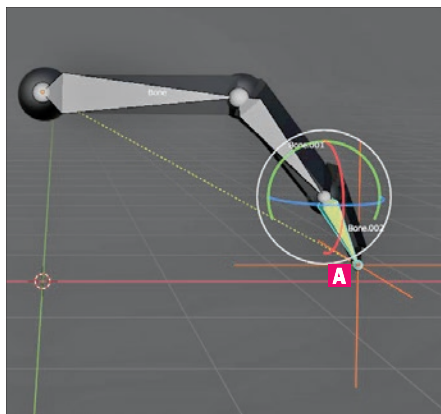
**1** Najpierw jednak musimy wprowadzić w **Pose Mode** coś, co się nazywa **Inverse Kinematic**. Zaznaczamy ostatnią kość. Następnie przechodzimy do zakładki **Pose** i wybieramy **Inverse Kinematic**, **Add IK to Bone**.




**2** Następnie wybieramy **To New Empty Object**.




**3** Spowoduje to nałożenie pewnych ograniczeń i połączeń. A także doda nowy obiekt, który będzie nam pozwalał na sterowanie nogą **A**.




**4** Przechodzimy do **Object Mode** i zaznaczamy ten nowy element. Gdy będziemy go przesuwać, noga, a właściwie stopa, podąży za nim. Pociągnie za sobą również resztę nogi, ale należy zauważyć, że początek zawsze zostaje w jednym miejscu.




## Dodajemy ciało do nogi

Dodamy teraz ciało, do którego będzie podłączona noga.

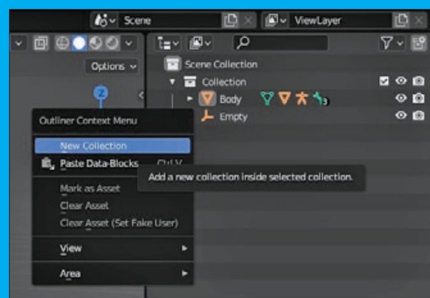
**1** Wśród plików na płycie znajdziemy plik o nazwie **Body.fbx**. Importujemy go tak, jak wcześniej nogę **A**.

**2** Po zaimportowaniu przesuwamy ciało i ustawiamy wraz z nogą. Pamiętajmy, że jak chcemy w **Object Mode** przesunąć całą

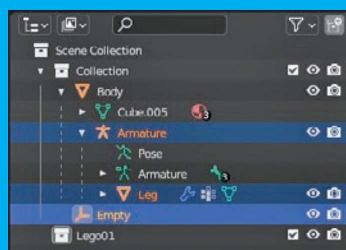
## PORZĄDKI

Zanim zaczniemy przygotowywać animację poruszania, zróbmy trochę porządku. Za jakiś czas będziemy kopiować nogi i przybędzie nam obiektów. By nad nimi zapanować, warto każdą nogę umieścić w innej kolekcji. Kolekcje pozwalają na sortowanie i grupowanie elementów sceny. Możemy je kopiować razem, wyłączać czy zablokować możliwość klikania.

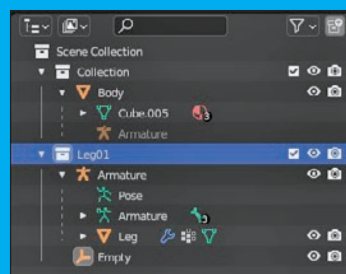
**1** W **Outlinerze** klikamy prawym przyciskiem myszy i gdy pojawi się lista, wybieramy **New Collection**.




**2** Nową kolekcję nazywamy **Leg01**. Zaznaczamy teraz w **Body Armature**, po rozwinięciu, również **Leg** i **Empty**.

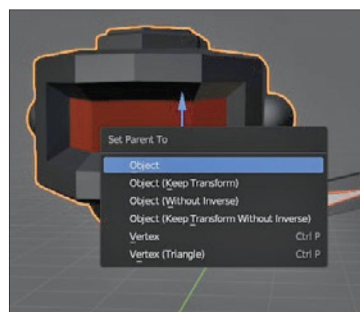


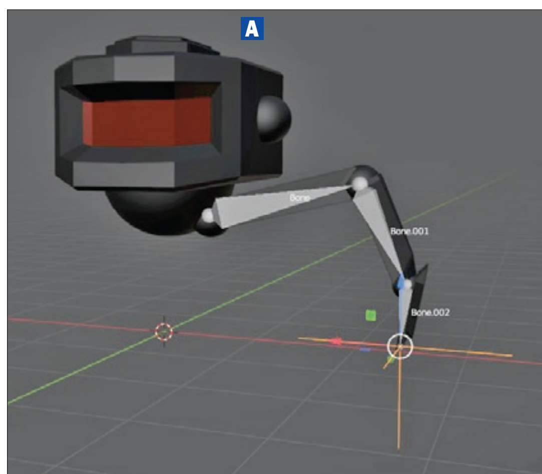

**3** Następnie przeciągamy je do kolekcji **Leg01**.

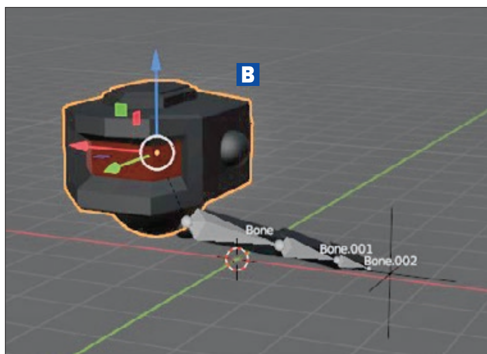



**Uwaga!** Ważne jest, by zaznaczyć również model, który jest ustawiony jako dziecko **Armature**. Pomimo połączenia Blender mógłby nie zrozumieć, że model też ma się znaleźć w kolekcji.

nogę, to przesuwamy **Armature**. Połączymy nogę z ciałem. Dzięki temu po przesunięciu ciała noga podąży za nim. Zaznaczamy najpierw **Leg**, a potem **Body** i naciskamy ctrl+P.



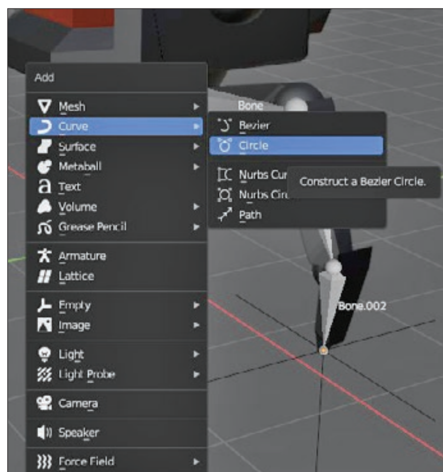





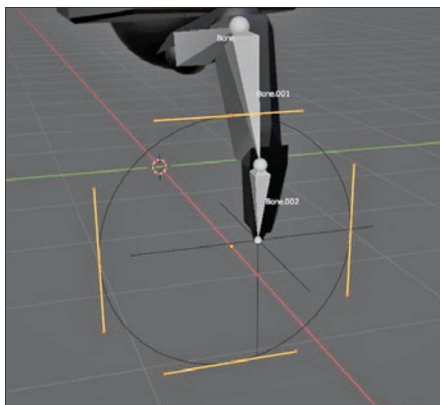

**3** Zobaczmy, co się stanie, gdy przesuniemy ciało **B**.

## Poruszanie nogą

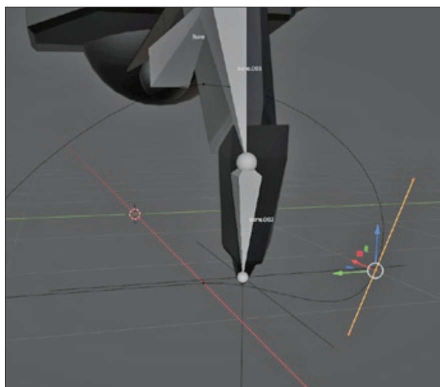
**1** Czas na poruszanie nogą. Musimy dodać krzywą w kształcie koła. Posłuży nam ona jako ścieżka, po której będzie obracać się noga. Wybieramy **Add**, **Curve**, **Circle**. Przesuwamy je w miejsce, gdzie znajduje się stopa.



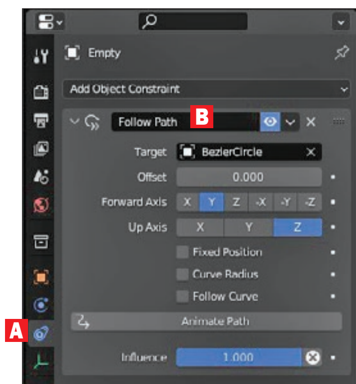

**2** Następnie przechodzimy do zakładki **Modeling**. Na sam początek obróćmy krzywą. Ważne jest, by zrobić to w **Edit Mode**, gdyż to pozwoli zachować kierunek krzywej. Najlepiej nacisnąć klawisze: R (rotate), Y (oś, po której obracamy), **-90** (obrót o tyle stopni, minus jest ważny).

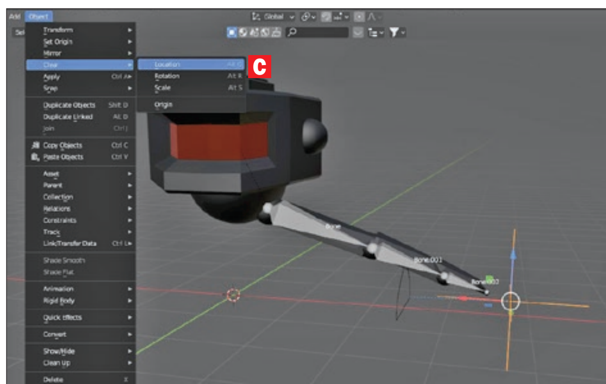


**3** Zmieniamy krzywą w półkole. Zaznaczamy środkowy, dolny wierzchołek i wyrównujemy. Po bokach zrobi nam się dziwne wykrzywienie. Możemy je wyrównać, klikając na boczne wierzchołki – zastosujemy tu inną kombinację klawiszy. Wybieramy: S (scale), Z (oś), **0**. Dzięki temu punkty pomocnicze wyrównają się z wierzchołkiem i uzyskamy równe zagięcie.



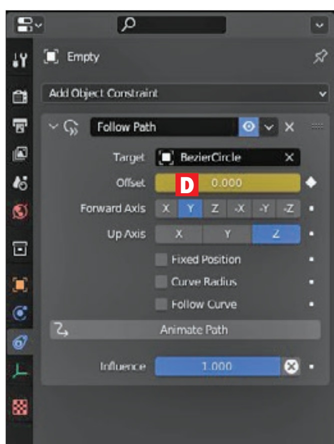
**4** Wracamy do zakładki **Layout**. Jeżeli kółko jest za duże, możemy je delikatnie zmniejszyć. Sprawdźmy też, czy nasza krzywa znajduje się w kolekcji **Leg01**. Jeżeli nie, to ją tam przesuńmy. Teraz wybieramy kontroler **Empty**. Przechodzimy do zakładki **Object Constraints Properties** **A** i dodajemy nowy **Constraint** – **Follow Path** **B**. Jako **Target** wybieramy krzywą.




**5** Noga może się dziwnie ustawić; to dlatego, że kontroler miał już wcześniej jakieś przesunięcie. Wybieramy więc **Object**, **Clear**, **Location** **C**. Dzięki temu noga powinna wrócić na swoje miejsce.

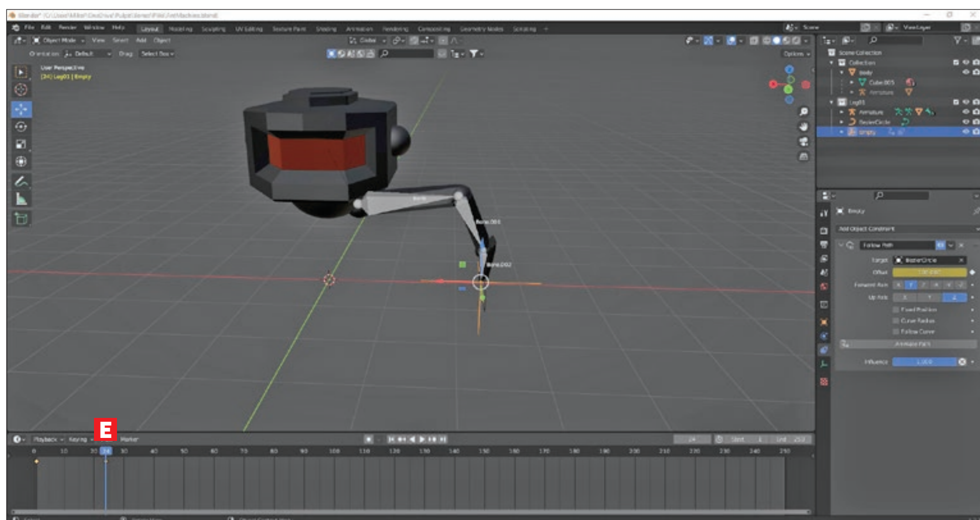
**6** Teraz na osi czasu zaznaczamy klucz dla wartości **Offset**. Ten parametr odpowiada przesunięciu na ścieżce. Cały przebieg to **100**, więc w pierwszej klatce ustawiamy **0** **D** i dodajemy klucz.




**7** Następnie przesuwamy się do klatki 24 **E** i tam ustawiamy **100** w pozycji **Offset** i ponownie dodajemy klucz.

**8** Dodajemy teraz jeszcze za pomocą skrótu klawiaturowego Shift+E na osi czasu **Linear Extrapolation**. Dzięki temu ruch będzie jednolity, a także będzie się przekładał na kolejne klatki.

Zobaczmy teraz, jak porusza się nasza noga.

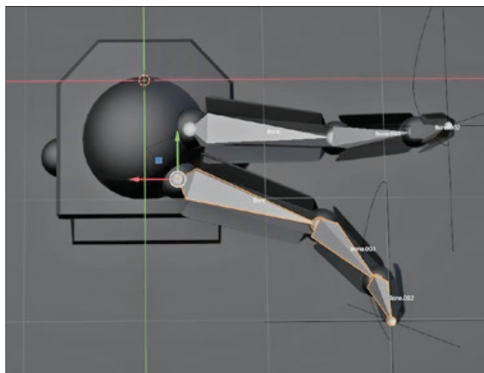


## Dodajemy kolejne nogi

Możemy przejść do dodawania kolejnych nóg.

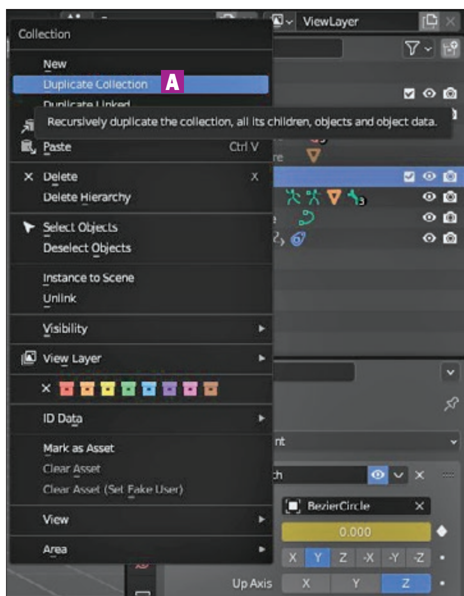
**1** Zatrzymujemy animację i przechodzimy do **Outlinera**. Zaznaczamy w nim kolekcję **Leg01**. Klikamy na nią prawym przyciskiem myszy i wybieramy **Duplicate Collection** **A**. Zmieniamy nazwę nowej kolekcji.

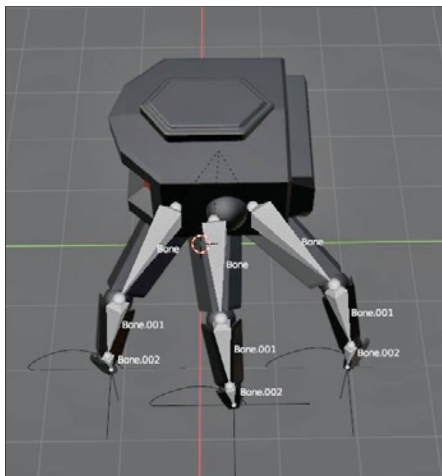
**2** Gdy mamy już utworzoną kopię, przesuwamy szkielet drugiej nogi, a następnie krzywą. Kontroler podąży za nią. Dzięki temu cała noga będzie przesunięta.





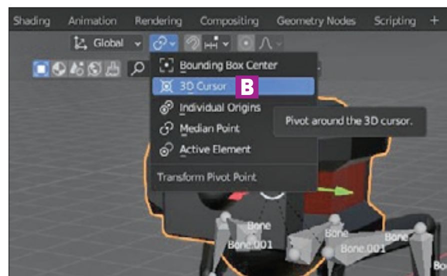
**3** To samo robimy z trzecią nogą, czyli: kopiujemy kolekcję, zmieniamy nazwę, przesuwamy kości i krzywą.




**4** Mamy teraz trzy nogi z jednej strony. Nasz model dla równowagi powinien mieć jednak nogi także z drugiej strony. Kopiujemy najpierw jedną z nóg. Teraz odbijemy model w taki sposób, by przesunął się dokładnie na drugą stronę. Sprawdzamy, czy **Body** w osi **X** jest na pozycji **0**. Jeżeli nie – wyrównujemy to. Teraz musimy ustawić środek dla zaznaczonych obiektów. W gór-

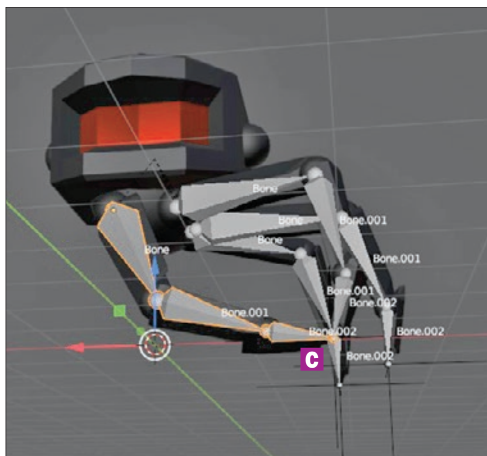
nym panelu znajdziemy listę **Transform Pivot Point**, która pozwoli nam na wybór.




**5** Wybieramy **3D Cursor** **B**, czyli środkiem transformacji będzie ten biało-czerwony obiekt. Zaznaczamy teraz kości naszej nogi i klikamy na **Object**, **Mirror**, **X Global**.

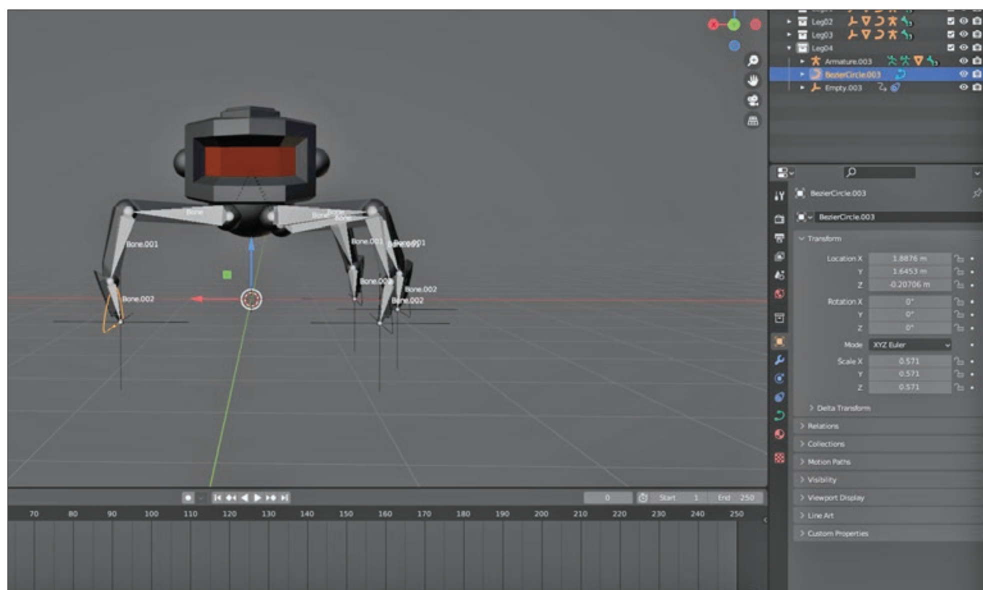


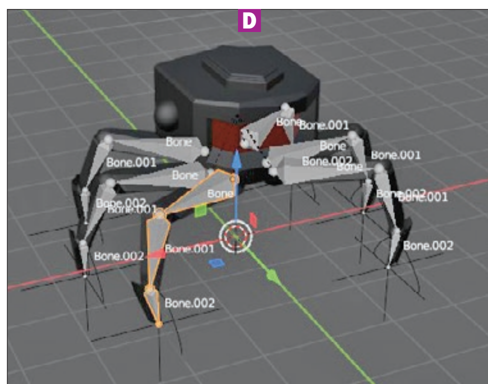

**6** Noga przesuwa się na drugą stronę robota. Jednak stopa zostaje na miejscu **C**.




Jest to spowodowane tym, że krzywa nie została przesunięta. W tym przypadku przesuwamy ją ręcznie.

**7** Możemy sprawdzić w **Object Properties** pozycję w osi **X** i ustawić na przeciwną, czyli usunąć lub dodać minus. (Nie przesuwamy krzywej za pomocą **Mirror**, gdyż później może to odwrócić oś w niepożądany sposób).



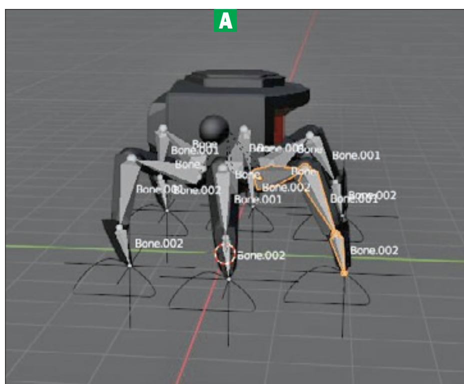



**8** Teraz robimy to wszystko dla pozostałych nóg **D**, czyli: kopiujemy kolekcję, zaznaczamy kości, odwracamy za pomocą **Mirror**, a na koniec przesuwamy krzywą.

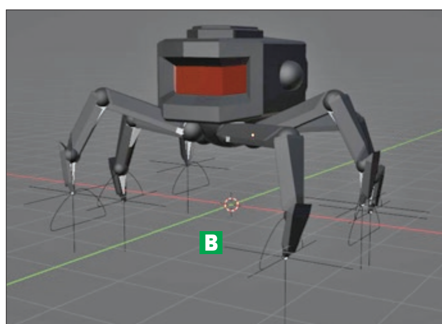
## Ruch całego robota

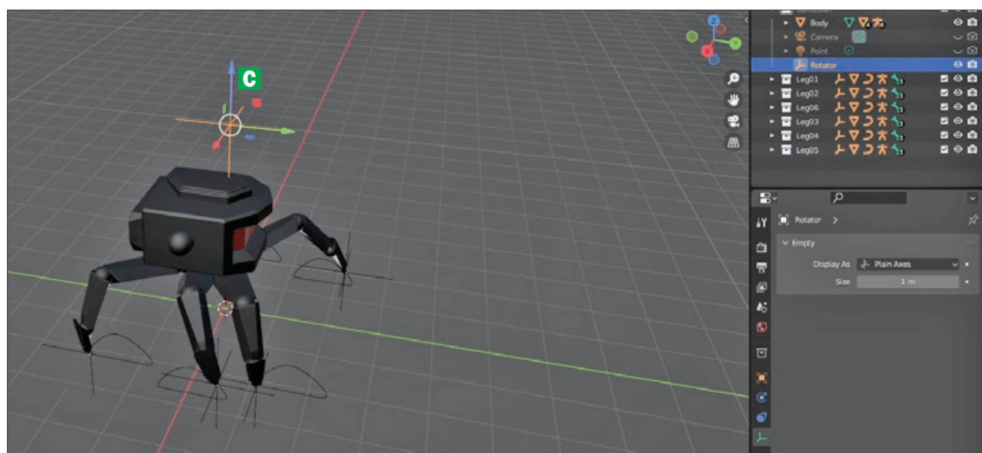
Nasz robot ma w końcu wszystkie nogi **A**. Jednak mają one tak samo ustawione klatki kluczowe. Powoduje to, że gdy włączymy animację, wydaje się, że robot skacze zamiast się poruszać normalnie. Musimy to naprawić.

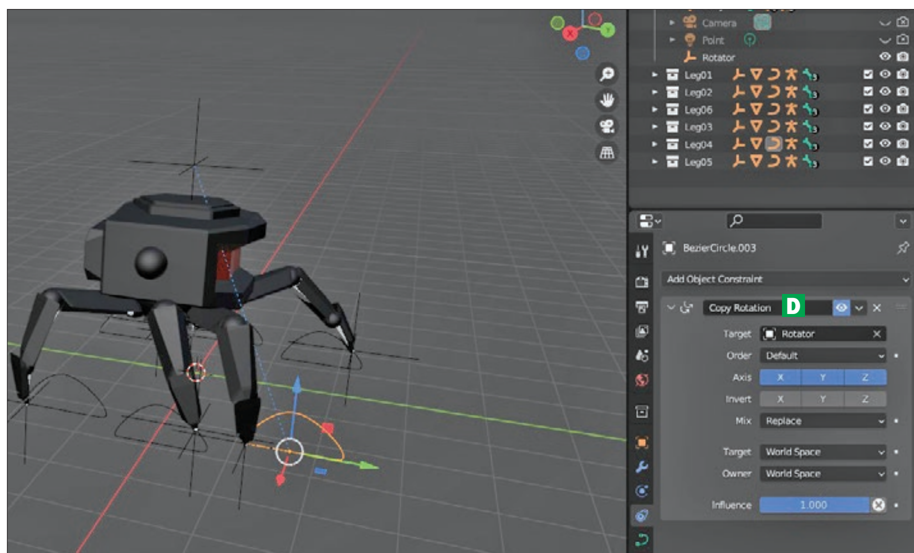
**1** By zapobiec skakaniu robota, po prawej stronie wybieramy przednią i tylną nogę, a właściwie ich kontrolery, a po lewej środkową **B**. Ich klucze na osi czasu przesuniemy o 12 klatek do przodu.




**2** Dzięki temu nogi będą się poruszały równomiernie. Zawsze robot będzie stał na co najmniej trzech nogach, co pozwoli mu zachować równowagę. W taki sposób właśnie poruszają się owady. A animatorzy i programiści często naśladują naturę.



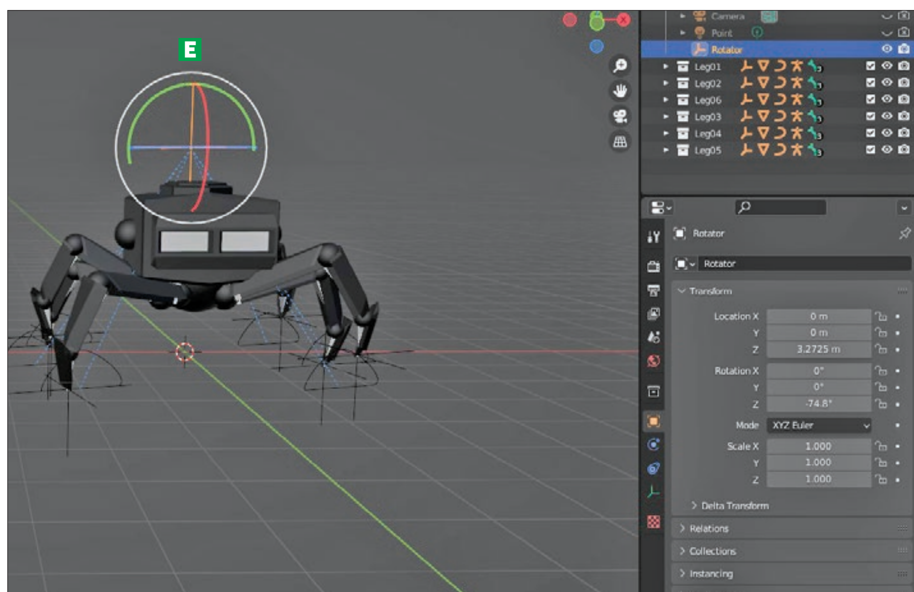



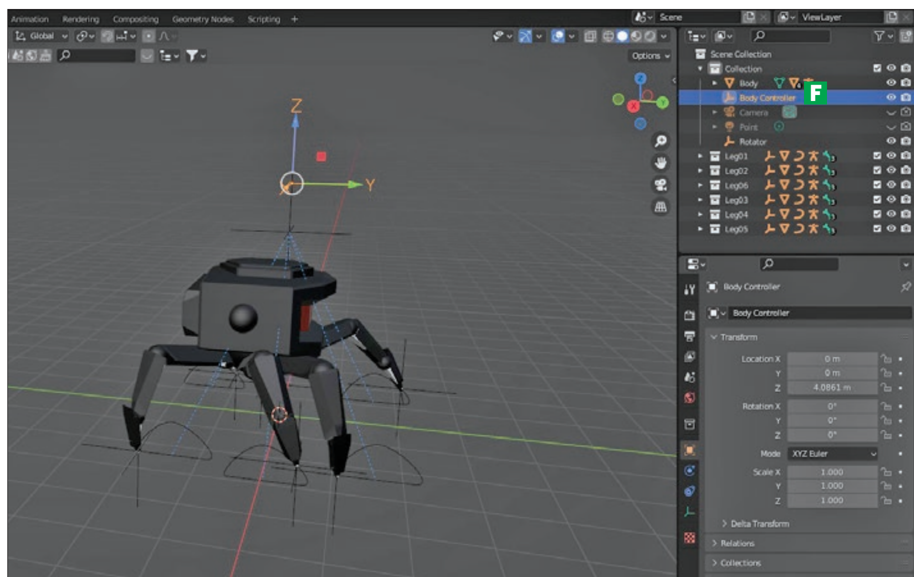



**3** Zadbajmy teraz o możliwość kontrolowania kierunku, w którym poruszają się nogi. Dodajemy kolejny, pusty obiekt, na przykład **Plain Axes**. Pamiętajmy, by znajdował się w kolekcji **Collection**. Nie jest on częścią nóg. Nazwiemy go **Rotator**. Ustawiamy go dokładnie nad naszym **Body** **C**.

**4** Przechodzimy teraz do pierwszej krzywej i dodajemy jej nowy **Constraint** o nazwie **Copy Rotation** **D**. Jako nasz **Target** wybieramy nasz **Rotator**.

**5** Robimy to dla wszystkich pozostałych krzywych. Teraz, gdy będziemy obracać






**Rotatorem** **E**, ścieżki też będą się obracać, a wraz z nimi zmieni się kierunek nóg.

**6** Zerujemy ten obrót i dodajemy kolejny **Empty Object**, na przykład: **Arrows**. Nazwiemy go **Body Controller** **F**.



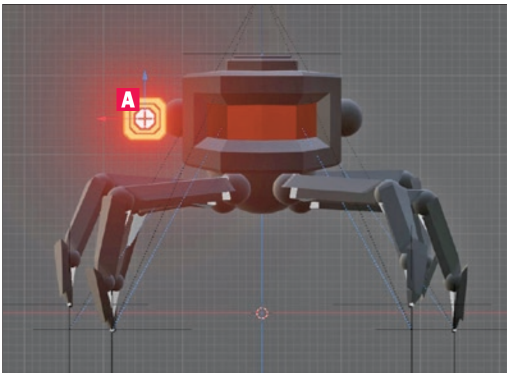

**7** Zaznaczamy teraz po kolei: **Body**, wszystkie krzywe, **Rotator** i na koniec **Body Controller**. Teraz korzystamy z ctrl+P i łączymy wszystkie nasze obiekty. Mamy teraz obiekt, który pozwoli kontrolować przesunięcie naszej postaci. Te czarne, przerywane linie **G** mówią, że elementy są z sobą w relacji.

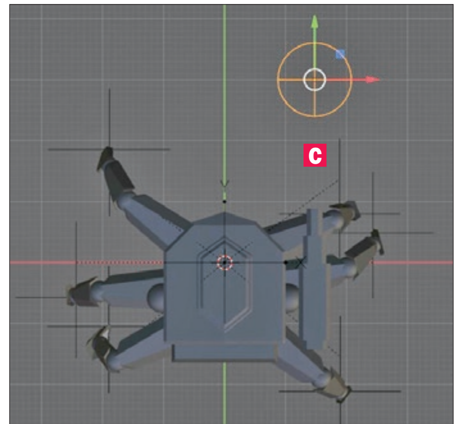
## Dodajemy broń dla robota

Dodajmy teraz ostatni element do naszego modelu – broń dla robota.

**1** Na płycie dołączonej do książki znajdziemy model broni – **Gun.fbx**. Importujemy go do naszego projektu.

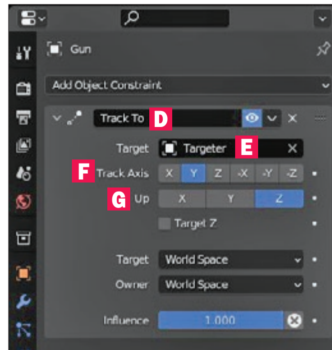




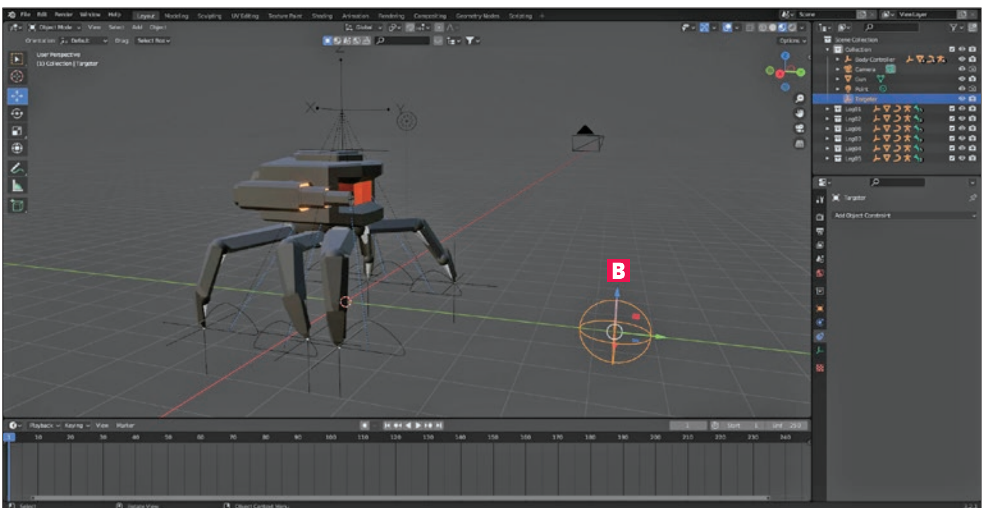

**2** Dodajemy broń koło **Body** A i ustawiamy tak, by stykała się z kulkami, które się tam znajdują.

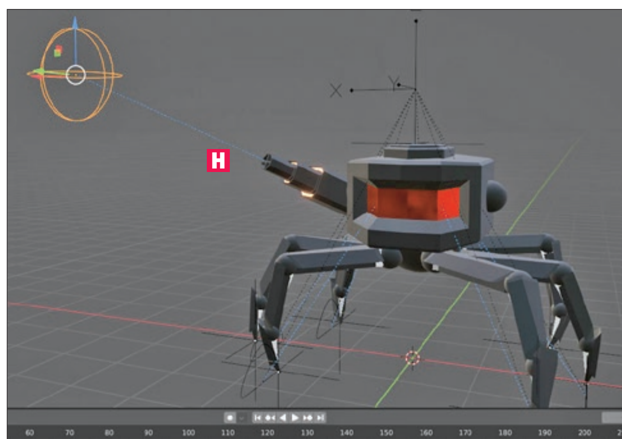
**3** Będziemy teraz chcieli spowodować, by broń mogła celować w różne miejsca, jednak z pewnymi ograniczeniami. Dodajemy pusty obiekt, na przykład sferę B. Posłuży nam jako element, w którego stronę zawsze będzie celować broń. Nazwiemy sferę **Targeter**.

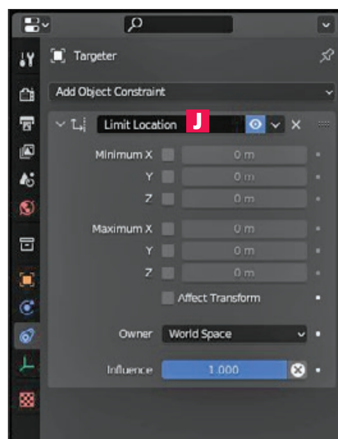



**4** Dobrze będzie ustawić **Targeter** w tej samej pozycji co nasza broń, tylko z przesunięciem na osi **Y** C. Kopiujemy więc pozycję broni i wklejamy ją do **Targetera**.

**5** Przechodzimy teraz do broni i dodajemy jej **Constraint**, który pozwoli jej zawsze podążać za celownikiem. Nazywa się **Track To** D. Jako **Target** wybieramy **Targeter** E, jako **Track Axis** F musimy wybrać **Y**, a obok **Up** G zaznaczamy **Z**.



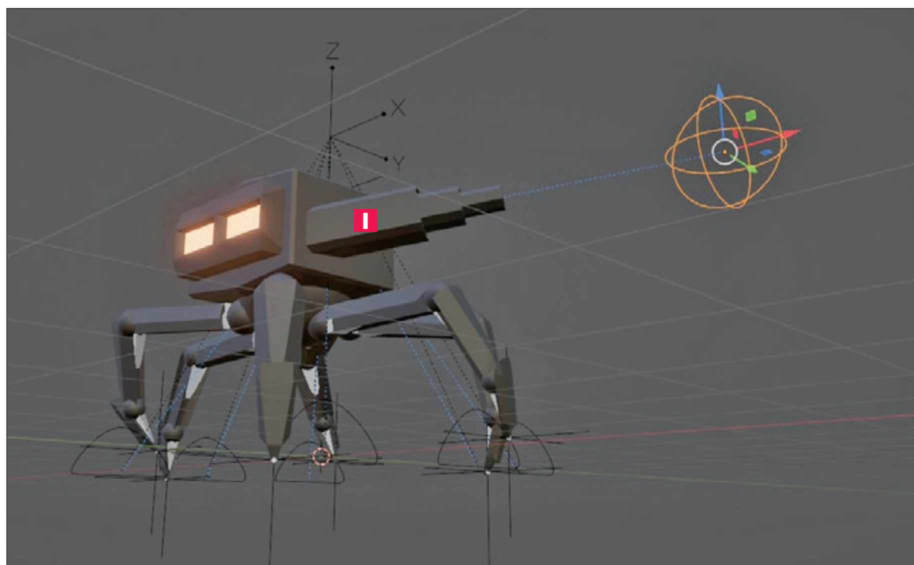


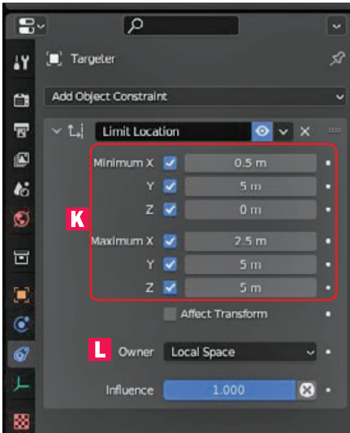



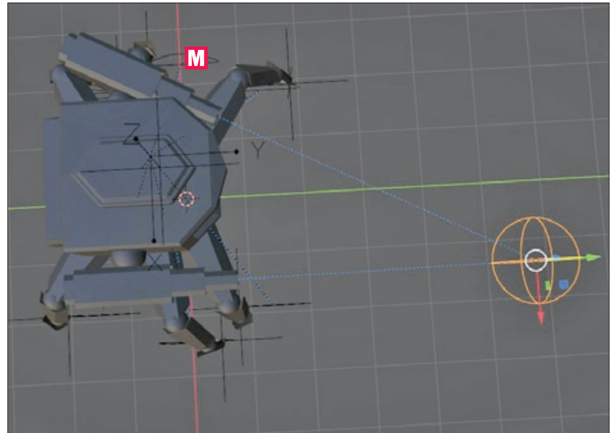

**6** Teraz, gdy będziemy przesuwać celownik, broń podąży za nim **H**.

**7** Pojawia się jednak pewien problem. Gdy za bardzo przesuniemy celownik, broń obraca się tak, że wchodzi w inną część robota **I**. Możemy temu zapobiec. Resetujemy ustawienia celownika. Dodajemy do niego **Constraint** o nazwie **Limit Location** **J**. Pozwoli on nam na ograniczenie ruchu celownika.

**8** Parametry **Minimum** i **Maximum** **K** określają, jak bardzo od pozycji 0 można przesuwać obiekt, do jakich pozycji możemy ograniczyć celownik. Najlepiej sprawdzać w **Object Properties**, w których miejscach broń nachodzi na robota. Najważniejsze to zmienić **Owner** **L** z **World Space** na **Local Space**. Dzięki temu środek, od którego będzie wyliczana pozycja broni, będzie mobilny i niezależnie od tego, gdzie będzie robot, będziemy mogli odpowiednio przesunąć celownik.



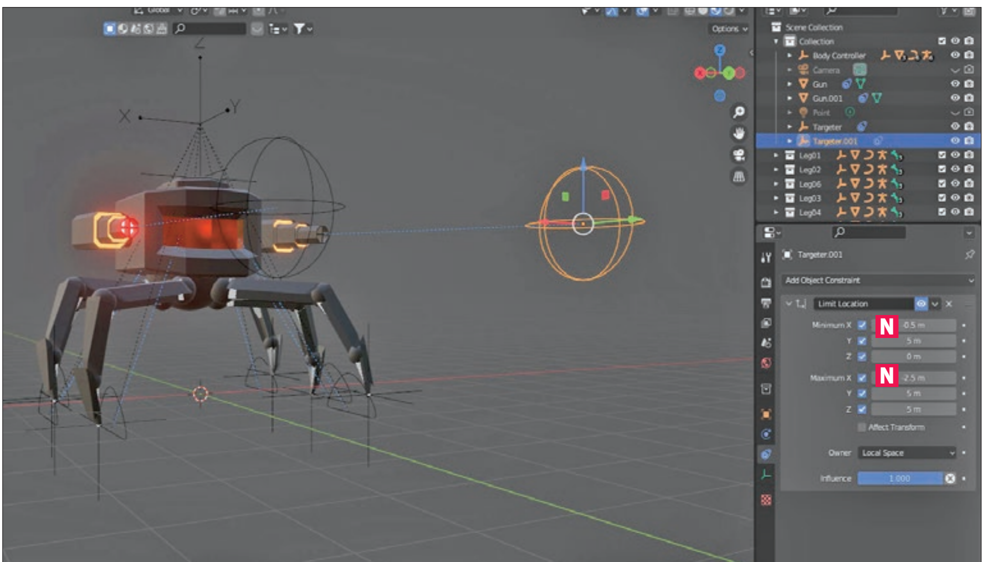



**9** Możemy teraz zduplikować naszą broń wraz z jej celownikiem i ustawić z drugiej strony **M**. Jako że drugi celownik ma te same ograniczenia co pierwszy, zablokuje się i druga broń będzie źle funkcjonowała.

**10** By to naprawić, wystarczy, że zmodyfikujemy ograniczenia na osi, która jest zablokowana. W tym przypadku wystarczy dopisać minusy w **Limit Location** w pozycjach **X** **N**.

**11** Teraz trzeba połączyć nasze bronie z resztą robota. Zaznaczamy najpierw je, a potem **Body**, naciskamy ctrl+P i wybieramy **Object**. Broń będzie połączona z kokpitem, natomiast celowniki podłączymy do **Body Controllera**. Teraz nawet jak wszystko przesuniemy, to dalej będziemy mogli manewrować celownikami. Zobaczmy jeszcze, co by się stało, gdybyśmy w **Limit Location Owner** zostawili wartość **World Space**. Celowniki w pewnym mo-





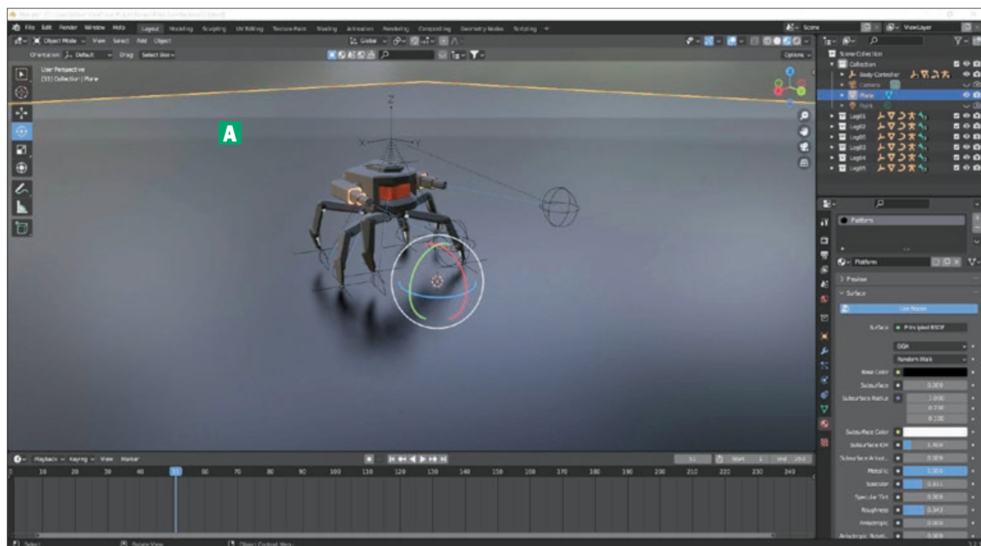

mencie zatrzymałyby się w miejscu, zupełnie zaburzając obroty 0.

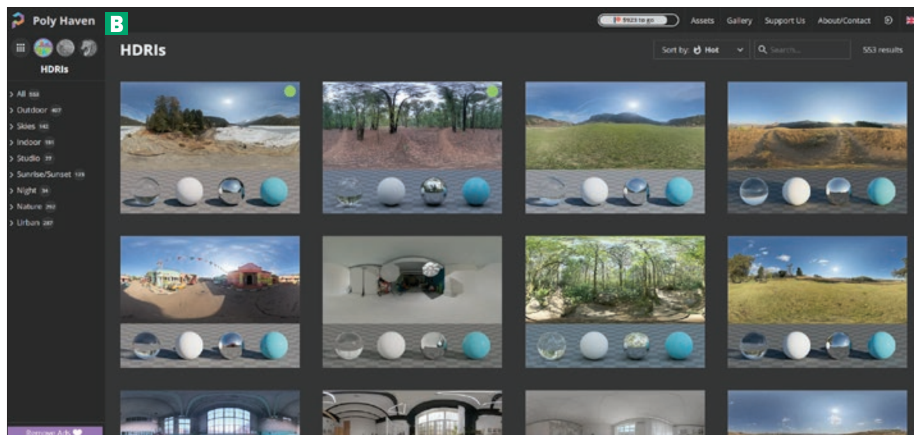
## Dodajemy scenę

**1** Czas na ostatnie szlify. Dodajemy obiekt **Plane**, który posłuży nam za podłoże A. Poszerzamy je. Następnie przydałoby się dodać jakiś materiał.

**2** W **Render Properties** załączamy **Ambient Occlusion**, **Bloom** i **Screen Space Reflection**. Dodajmy teraz trochę światła zewnętrznego.

**3** Wchodzimy na stronę **polyhaven.com** B. Jest to strona, na której można znaleźć wiele darmowych assetów do naszych

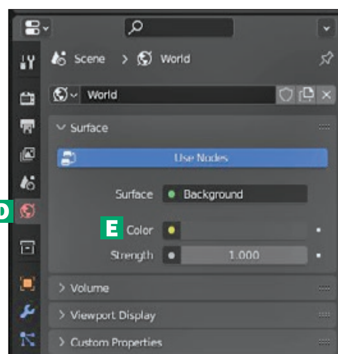




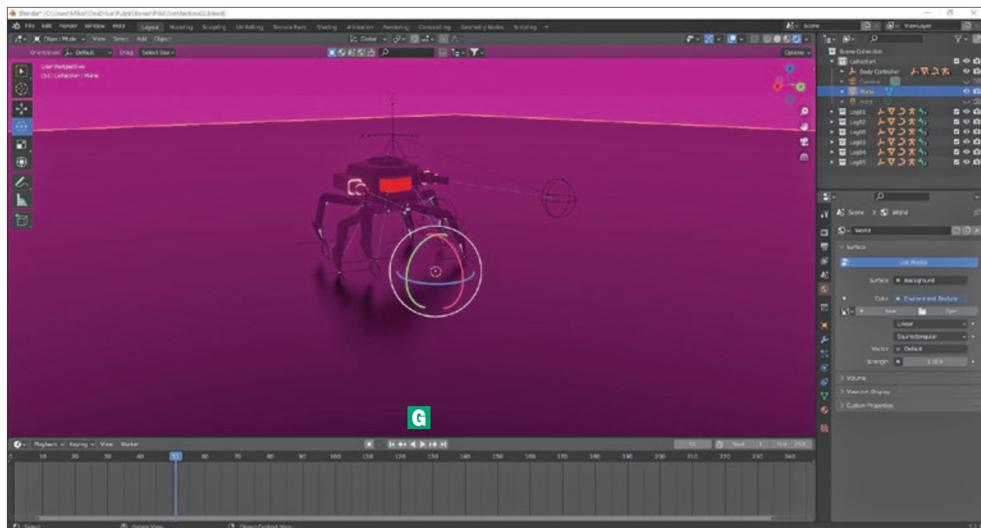

projektów. Nas będą interesowały obrazy **HDRI**. Są to specjalne tekstury, które pozwolą nam imitować oświetlenie z otoczenia. Klikamy na **Browse HDRIs** u dołu strony i szukamy tekstury, która nam się spodoba i będzie pasowała do projektu.

**4** Gdy już jakąś wybierzemy, klikamy na nią. Na górze po prawej znajdziemy przycisk, który informuje o rozdzielczości tekstury oraz o jej formacie. Standardowo **4K** powinno wystarczyć. Jeśli potrzebujemy, możemy wybrać większą, pamiętajmy tylko o tym, że im wyższa rozdzielczość, tym więcej waży plik. Klikamy na **Download** **C** i czekamy, aż się pobierze.

**5** Przechodzimy do **World Properties** **D**. Tam znajdziemy pozycję **Color** **E** z żółtą kropką.

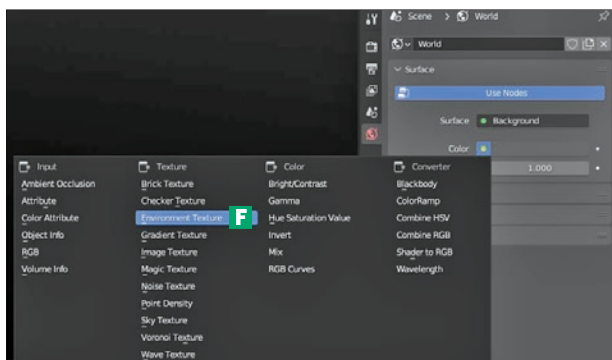


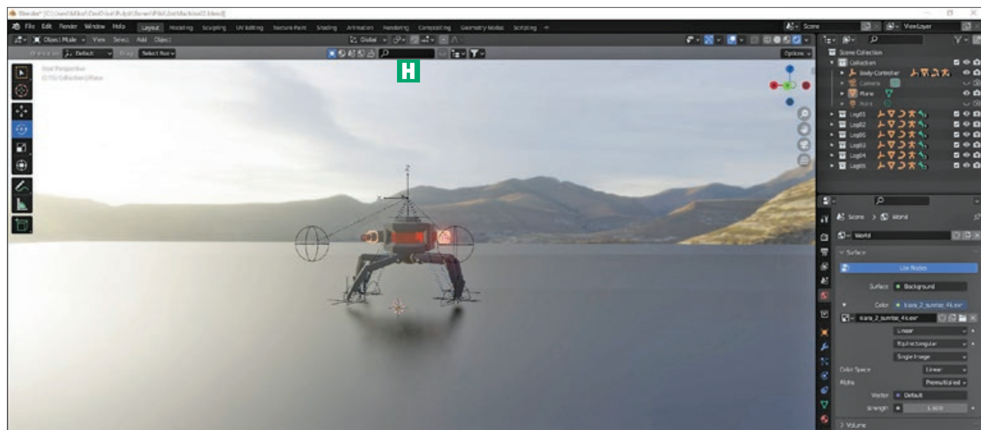



**6** Gdy na nią klikniemy, rozwinie się lista, z której musimy wybrać **Environment Texture** **F**, czyli teksturę otoczenia. Scena powinna się teraz zrobić różowa **G** (musimy być w **Viewport mode Rendered** – prawy górny róg sceny).

**7** Klikamy na przycisk **Open**. Pojawi się okno, w którym wskazujemy pobraną teksturę i zatwierdzamy nasz wybór.






Oto efekt **H**. Tekstura powinna się nałożyć na scenę, dodając światło, które ma w sobie zapisane.

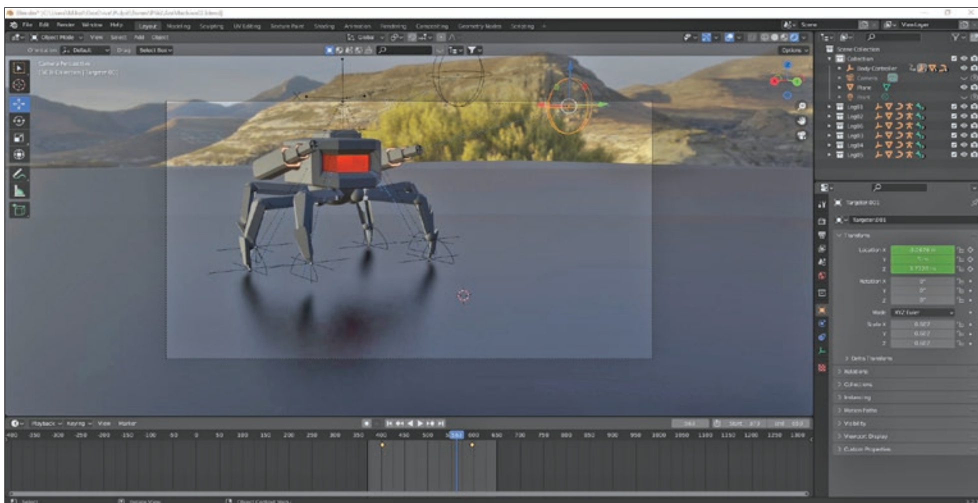
## Jak wprawić w ruch robota

Ostatnia rzecz, jaka nam pozostała do zrobienia, to wprawić całego robota w ruch. Można to zrobić tak jak z naszą lokomotywą w jednym z wcześniejszych projektów. Ale teraz poznamy prostszy sposób.

**1** Przesuwamy robota na koniec platformy i dodajemy do **Body Controllera** **A** klucz dla pozycji na osi czasu. Następnie przesuwamy go na koniec i dodajemy kolejny klucz na osi czasu. **Uwaga!** Warto wybrać odległą, ostatnią klatkę, by ruch nóg i ciała był współmierny.

**2** Oczywiście nie trzeba renderować wszystkich klatek. Możemy ustawić klucze na osi na bardzo odległych klatkach, a wyrenderować tylko fragment, gdy robot wchodzi zasięg kamery i aż do chwili, kiedy znika z drugiej strony.

**3** Możemy też dodać animację poruszania się broni. Dzięki kontrolerom będzie to znacznie ułatwione. Najważniejsze to dobrze się przy tym bawić.



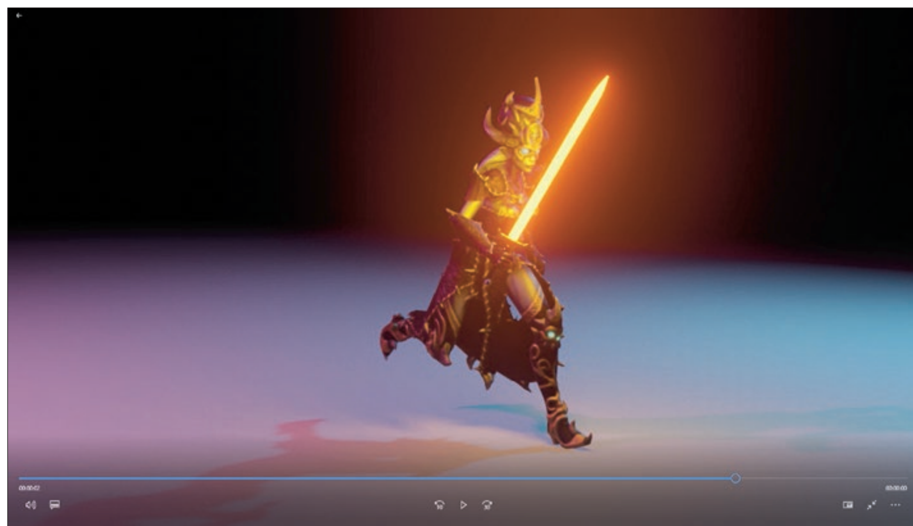
# 5 Blendowanie animacji



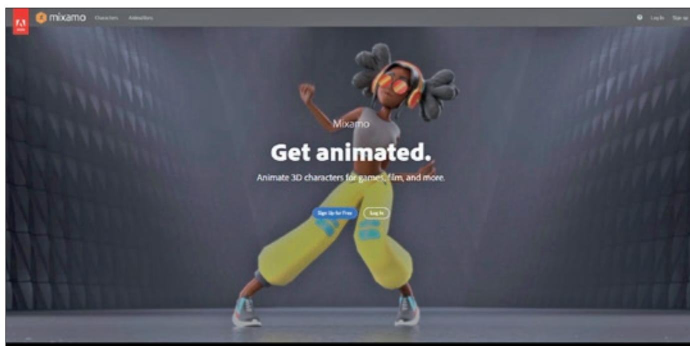

Na DVD znajdziemy pliki projektów przedstawionych w tym rozdziale

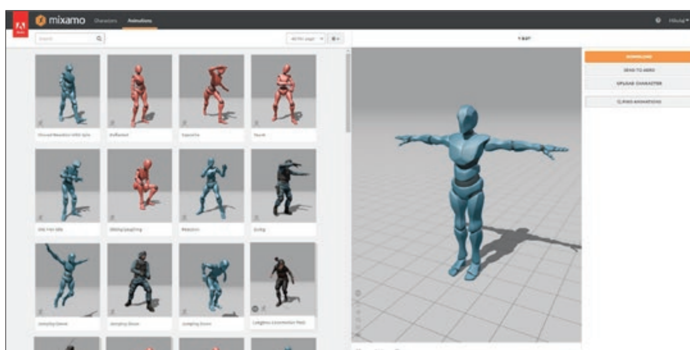
**Podczas pracy z animacjami często powstaje co najmniej kilka osobnych elementów, które następnie trzeba połączyć, czyli zblendować. Zobaczmy, na czym to polega**

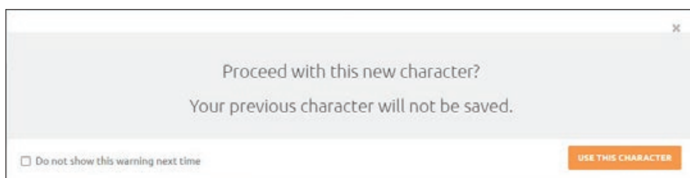
W tym rozdziale przeczytamy, w jaki sposób łączyć animacje w jeden ciąg. Czynność tę nazywamy **blendowaniem** – to mieszanie kilku różnych animacji.
Wiele współczesnych animacji to animacje postaci ludzkich. Ich przygotowanie zajmuje dużo czasu. Graficy, bazując na szkicach koncepcyjnych, przygotowują wiele postaci do gier czy filmów. My jednak możemy pobawić się animacją humanoidów, nie poświęcając czasu na modelowanie. By poćwiczyć, warto skorzystać z gotowców. W tym rozdziale będziemy animować gotową postać, łącząc dwa rodzaje ruchu, dodamy jej też miecz.

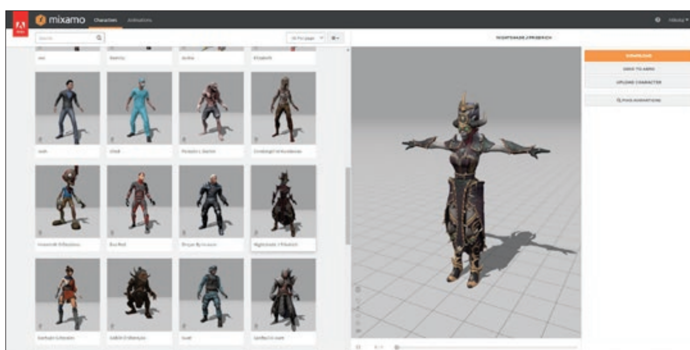


# Mixamo – pobieramy szablony



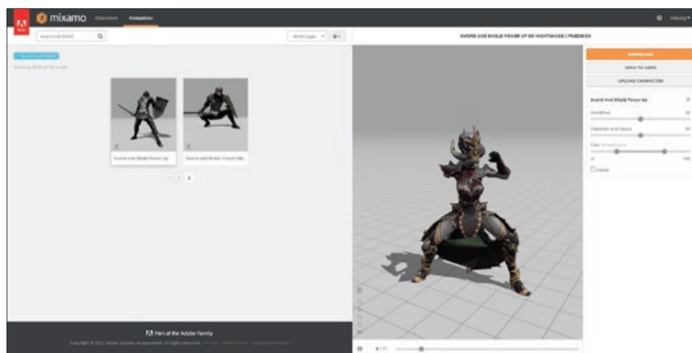

Gotowe szablony postaci znajdziemy na przykład na stronie **Mixamo** – pod adresem **mixamo.com**. Jest to projekt firmy Adobe. Wystarczy się zarejestrować, by móc za darmo korzystać z dużej bazy modeli i animacji. Jeżeli mamy już jakiś własny gotowy model, możemy go załadować, a platforma sama go zrigguje (o riggowaniu – patrz ostatni projekt w poprzednim rozdziale).

**1** Wybieramy z zakładki **Characters** jakąś postać. Mixamo zapyta, czy chcemy zamienić bieżący model na inny. Zatwierdzamy przyciskiem **Use This Character**.

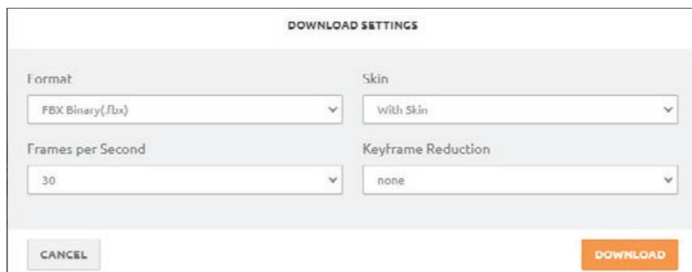
**2** Powinien się teraz pojawić podgląd. Możemy go obejrzeć z różnych stron. By go przesunąć, klikamy na niego i poruszamy myszą. Naciskając kółko myszy i poruszając nią, przesuniemy model w płaszczyźnie, a kręcąc kółkiem, przybliżymy lub oddalimy.

**3** Przechodzimy teraz do zakładki **Animations**. Tu możemy wybrać jedną z animacji i zobaczyć, jak prezentuje się ona na naszym

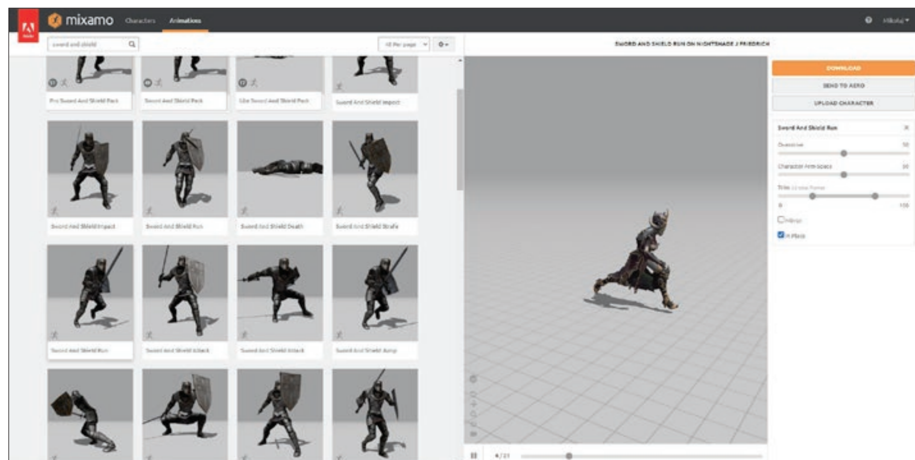
modelu. Do naszego projektu zastosujemy najpierw animację **Sword and Shield Power Up**. Możemy ją wyszukać za pomocą pola w lewym, górnym rogu.



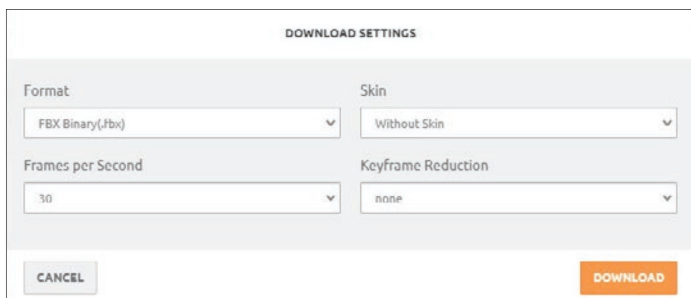
**4** Gdy mamy już wybraną animację, klikamy na **Download**. Musimy teraz określić parametry pobierania. Jako format wybieramy standardowy **FBX Binary**. W polu **Frames per Second** ustawiamy **30**. Natomiast w polu **Skin** wybieramy **With Skin**. Ten parametr odpowiada za to, czy animacja będzie pobierana wraz z modelem, czy bez niego. Pierwszą animację pobieramy z modelem, by mieć jakąś bazę. Kolejne już bez modelu, bo potem potrzebna nam już będzie sama animacja.




**5** Wybieramy teraz animację biegu. Najlepiej, by była z tej samej kategorii, czyli **Sword and Shield Run**. Gdy ją wybierzemy, w panelu po prawej suwakami określamy, jaka część animacji jest nam potrzebna (**Trim**) i jakiej potrzebujemy szerokości ustawienia ramion (**Character Arm Space**). Pole **In Place** to parametr, który w przypadku animacji będącej ruchem w jakimś kierunku utrzymuje model w jednym punkcie. Zaznaczamy to



pole. Będziemy chcieli powielić animację biegu, pozostawienie tego pola niezaznaczonego doda nam więcej pracy.




**6** Pozostaje ustawić parametry pobierania. Teraz nie potrzebujemy już modelu. Możemy więc w polu **Skin** wybrać opcję **Without Skin**. Pozostałe parametry pozostają takie jak poprzednio.

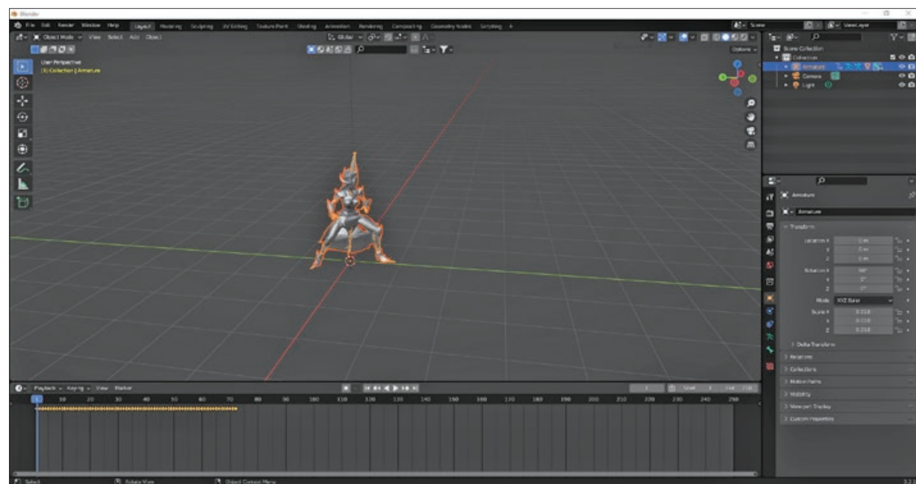


## Importujemy szablony do Blendera

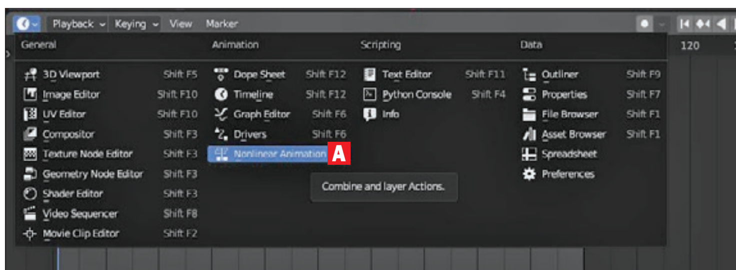
**1** Przechodzimy do Blendera. Usuwamy cube. Potem, klikając na **File**, **Import**, **FBX**, pobieramy pierwszą animację, tę z modelem.

**2** Możemy podejrzeć pobraną animację, klikając na **Play** na osi czasu.

**3** Importujemy drugą animację w taki sam sposób jak poprzednią. Na scenie pojawi się drugi szkielet, nałożony na poprzedni model.



# Przypisujemy animację do szkieletu



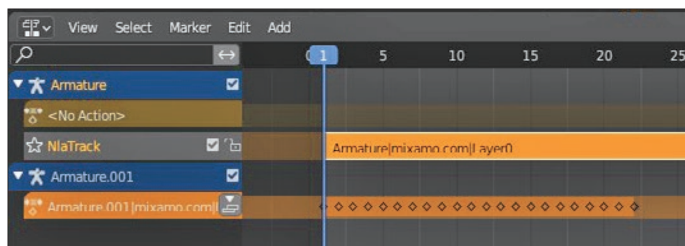

Będziemy teraz musieli przypisać animację z drugiego szkieletu do pierwszego. **Uwaga!** Pobierając animacje, należy zawsze wybierać animacje na tym samym modelu. Niektóre postaci mają dodatkowe kości; jeżeli drugi szkielet ich nie ma, może dojść do problemów.

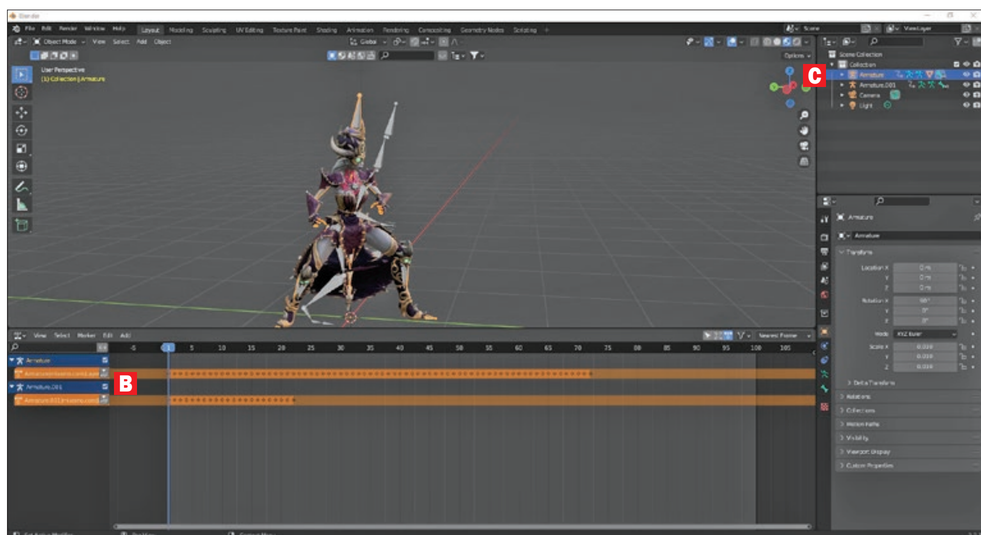
**1** Przechodzimy do panelu **Nonlinear Animation** (NLA) **A**, zastąpimy na jakiś czas **Timeline**.

**2** Na osi powinny się pojawić animacje szkieletów **B**. W **Outlinerze** zaznaczamy szkielet z modelem **C**. To ważne, by był zaznaczony.

**3** Gdy mamy już gotowe animacje, tworzymy nowe akcje. Klikamy na przycisk koło nazwy animacji.

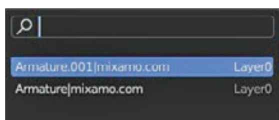
**4** Możemy usunąć drugi szkielet z całego projektu, nie będzie nam już potrzebny.

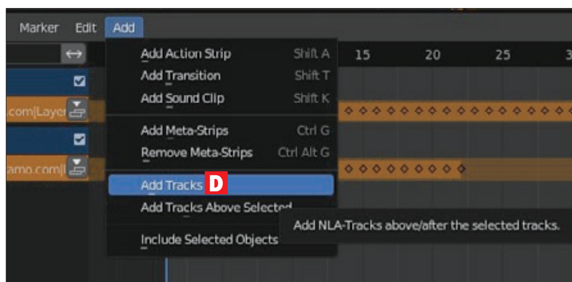


**5** Trzeba teraz dodać nową ścieżkę, klikając na **Add**, **Add Tracks** **D**.

**6** Następnie za pomocą [shift]+[A] dodajemy tę akcję, której szkielet wcześniej usunęliśmy.



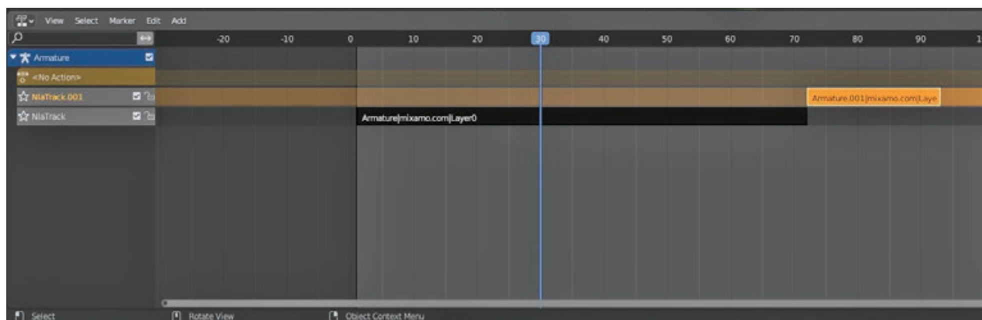

**7** Obie animacje powinny pojawić się na osi.



# Blendujemy i wypalamy animacje

**1** Naciskając klawisz spacji, możemy uruchomić i zatrzymać animacje. Gdy je włączymy, zauważymy jednak, że uruchamia się tylko jedna z nich. Jest to spowodowane tym, że może być uruchomiona tylko jedna naraz.

**2** Przesuńmy więc drugą animację (w naszym przykładzie – biegu) tak, by zaczynała się zaraz po skończeniu pierwszej. Niestety, dalej nie działa odtwarzanie ich po kolei.

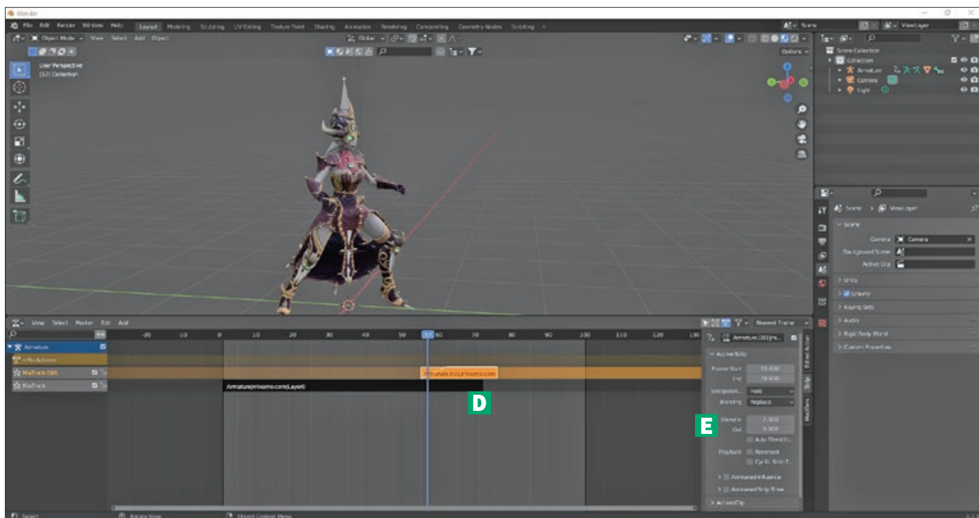


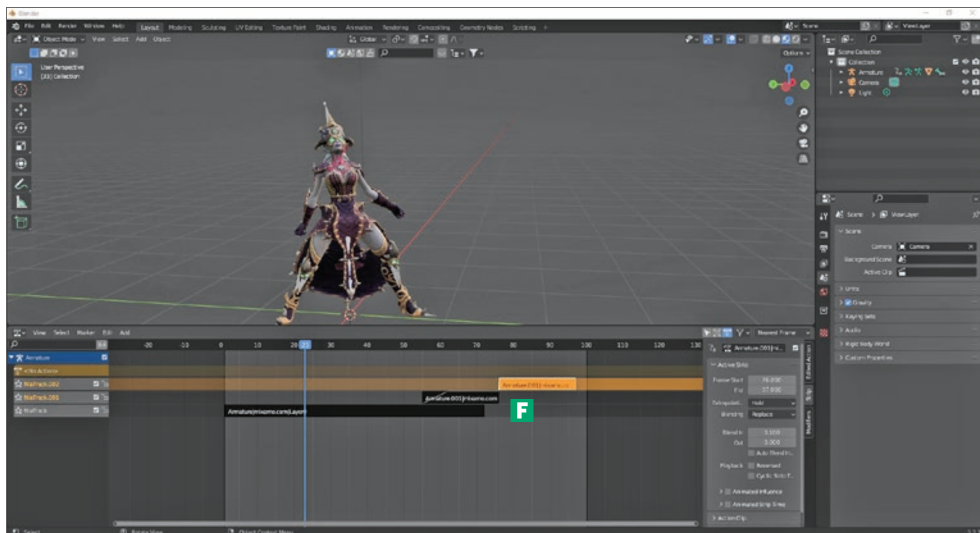



**3** Wyświetlamy prawy boczny panel **A** za pomocą klawisza N na klawiaturze.

**4** Zaznaczamy drugą animację i przechodzimy do zakładki **Strip** **B**. Znajdziemy tam parametr o nazwie **Blend In** **C**. Odpowiada on za mieszanie animacji. Ponieważ kości są te same, Blender może animacje z sobą pomieszać, uwzględniając pozycje w pierwszej i drugiej animacji. Jest to tak zwane blendowanie. Na przykład w grach, kiedy mamy animację chodzenia i biegania, gwałtowne przejście między nimi mogłoby spowodować dziwny skok ruchu. Dlatego programiści stosują takie mieszanie, by przejścia między animacjami wyglądały płynnie. W parametrze **Blend In** ustawiamy wartość **0.1**. Dzięki temu nasze






animacje powinny się teraz odtwarzać jedna za drugą.

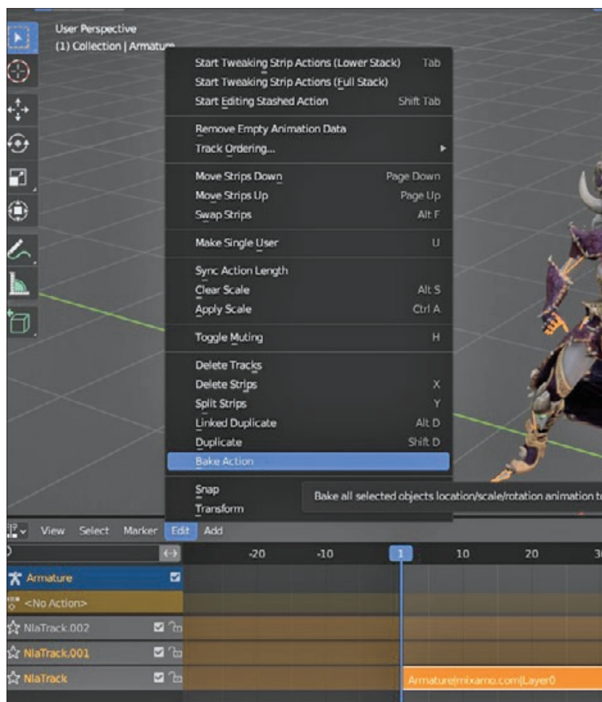
**5** Jednak widzimy właśnie ten skok. Najpierw kończy się jedna animacja, a dopiero potem uruchamia się druga. Przesuwamy więc bieg tak, by nachodził na poprzednią animację **D**, i zwiększamy wartość **Blend In** **E** (w przykładzie – do **7**). Poustawiajmy to tak, by efekt wyglądał jak najbardziej naturalnie.

**6** Pozostaje nam dodać jeszcze jedną akcję biegu. Zduplikujmy ją, a następnie przesuńmy **F**. W tym wypadku ustawimy akcję jedna za drugą, a blendowanie – na jak najmniejsze.

## Wypalamy animację

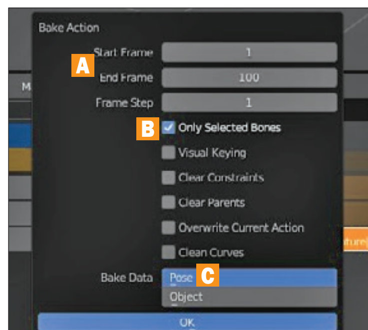
Możemy teraz wypalić naszą animację. Wypalanie polega na stworzeniu jednej animacji na bazie wszystkich akcji. Nie możemy jej już wtedy edytować, ale łatwiej jest na przykład eksportować ją do gry.

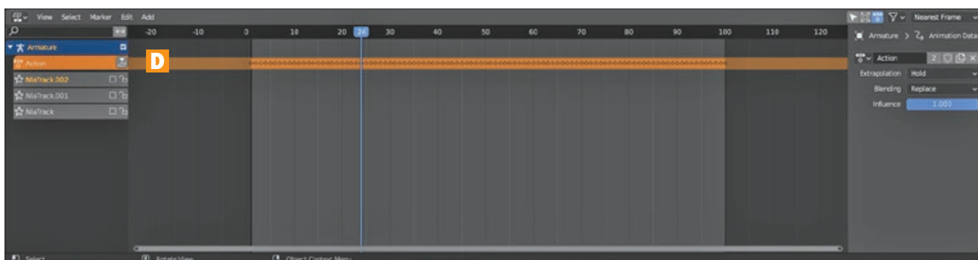
**1** Zaznaczamy szkielet, a następnie wszystkie akcje. Wybieramy **Edit**, **Bake Action**.



**2** Pojawia się okno. Wybieramy w nim, od której do której klatki chcemy wypalać **A**. Zaznaczamy **Only Selected Bones** **B** i przy **Bake Data** wybieramy **Pose** **C**.

**3** Po wypaleniu możemy zobaczyć nową akcję **D** złożoną z klatek kluczowych. Te, na których pracowaliśmy, możemy usunąć.



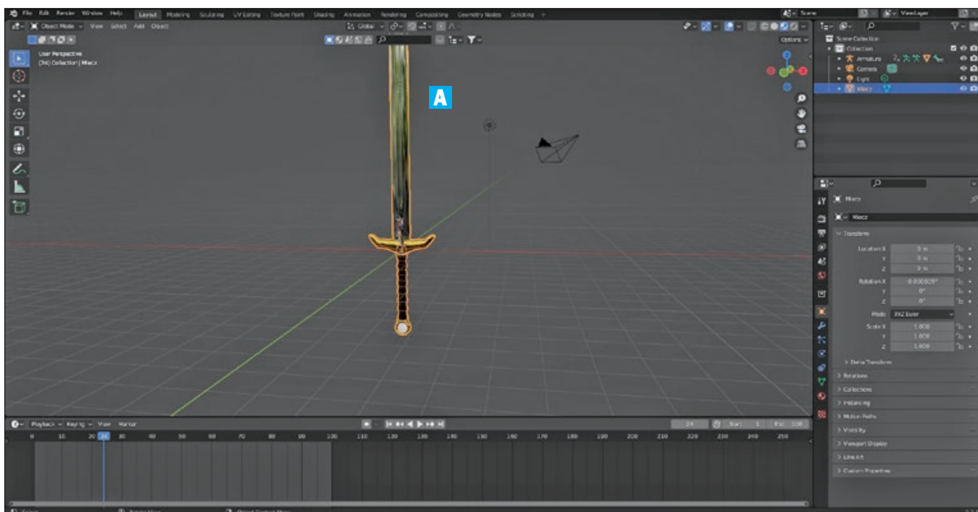

# Dodajemy postaci miecz

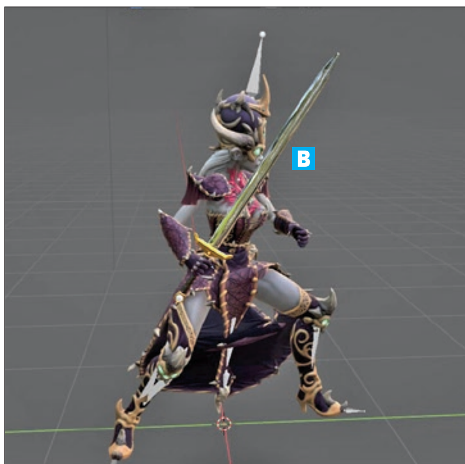
**D**odajmy teraz miecz. Włożymy go w rękę naszej naszej animowanej w Blenderze postaci.

**1** W oknie programu Blender kliknięciem wracamy do **Timeline**.

**2** Na płycie dołączonej do książki znajduje się plik **Sword.fbx**. Importujemy go do naszego projektu **A**.

**3** Zmniejszmy miecz do rozmiarów odpowiadających naszemu modelowi. Może-









my ustawić postać w wybranej pozycji, przesunąć miecz do ręki i zobaczyć, jaki rozmiar będzie najlepszy B.

**4** Gdy mamy to gotowe, możemy wyłączyć podgląd modelu postaci C.

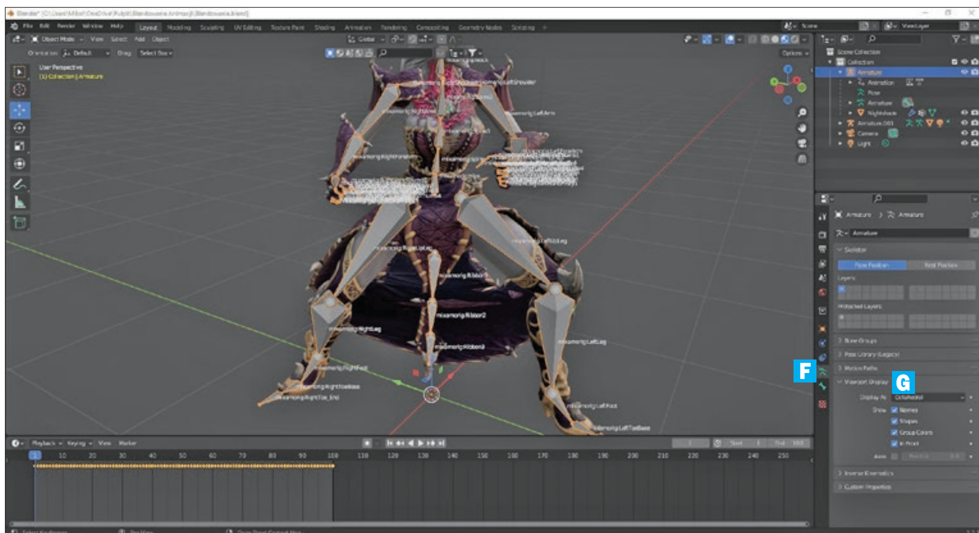
**5** Dodamy teraz obiekt – światło. Powinno to być światło typu **point**. Ustawiamy je na środku klingi D, a następnie łączymy je z nią za pomocą ctrl+P; z listy wybieramy **Object**. Pamiętajmy, że to światło musi być dzieckiem miecza.

**6** By dodać naszej postaci broń, musimy stworzyć kość dla miecza. Pozwoli to połączyć ją z kośćmi postaci. Tworzymy nową pozycję – **Armature**. Nazwijmy ją **Sword-Bone**. Ustawiamy ją tak, by początek kości znajdował się w miejscu, w którym nasza postać będzie chwytać miecz E.

**7** Teraz, tak samo jak łączyliśmy światło, łączymy nasz miecz z kością. Naciskamy ctrl+P i wybieramy **Object**. Kość ma być rodzicem miecza. Umieszczamy miecz z powrotem w ręce postaci.





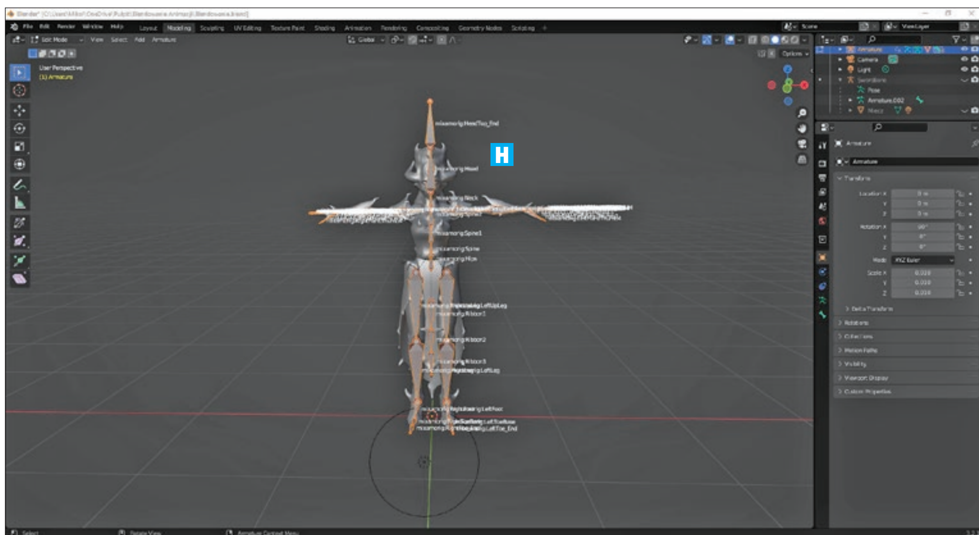



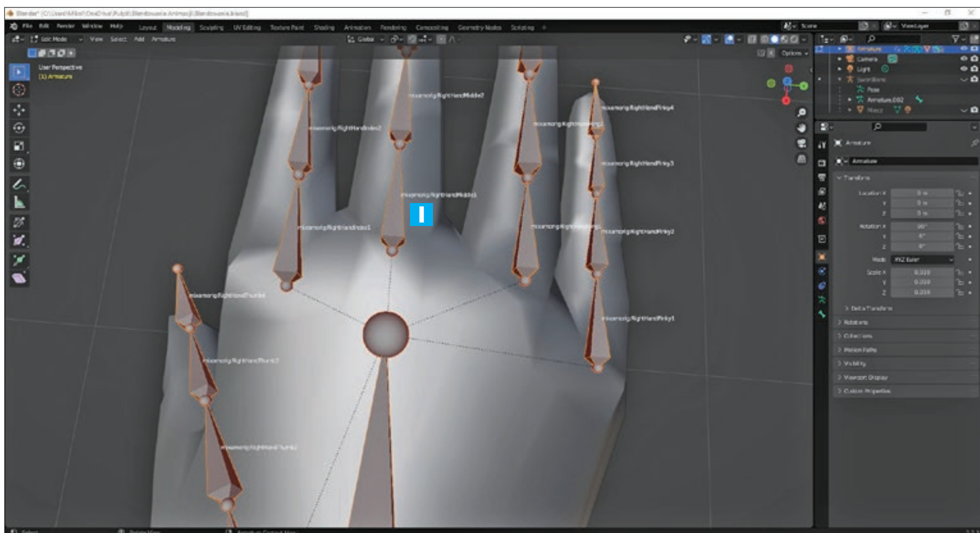

**8** Zaznaczamy jej kości i wybieramy zakładkę **Object Data Properties** **F**, a następnie w **Viewport Display** **G** zaznaczamy przy pozycji **Show** pola **Names** oraz **In Front**.

Gdy będziemy w **Edit Mode**, nasza postać zaprezentuje się w tak zwanym **T-Pose** **H**. Jest to jedna z podstawowych pozycji do riggowania postaci ludzkich.

**9** Przechodzimy teraz do zakładki **Modeling**. Będzie nam potrzebna nazwa kości, do której podłączymy kość z miecza.

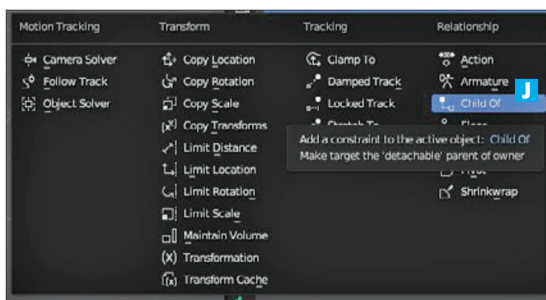
**10** Musimy teraz poszukać kości w prawej ręce. Najlepsza będzie pierwsza kostka środkowego palca. Poszukajmy jej



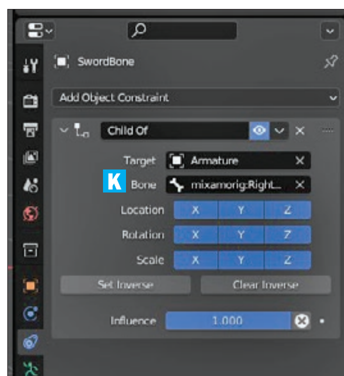



nazwy (**mixamorig.RightHand-Middle1** I).

**11** Zapamiętajmy tę nazwę. Przechodzimy z powrotem do zakładki **Layout**. Klikamy na **SwordBone** i przechodzimy do zakładki **Object Constraint Properties**. Dodajemy element **Child Of** J.




**12** Teraz jako **Target** ustawiamy szkielet postaci. Pojawi się pole **Bone** K i w nim wybieramy potrzebną nam kość.




**13** Niestety, może się zdarzyć, że miecz jest przesunięty. By to naprawić, wy-



bieramy **Set Inverse** w **Child Of**. Kość powinna wrócić na miejsce, a miecz wraz z nią **L**. Teraz, gdy uruchomimy animację, miecz będzie się poruszał wraz z ręką.
**Uwaga!** Może się zdarzyć, że miecz wchodzi w głowę postaci. Na tym etapie najlepiej pomniejszyć miecz. A w przyszłości, pobierając kolejne elementy, warto zwiększyć **Character Arm Space** w Mixamo.




# Ruch całej postaci

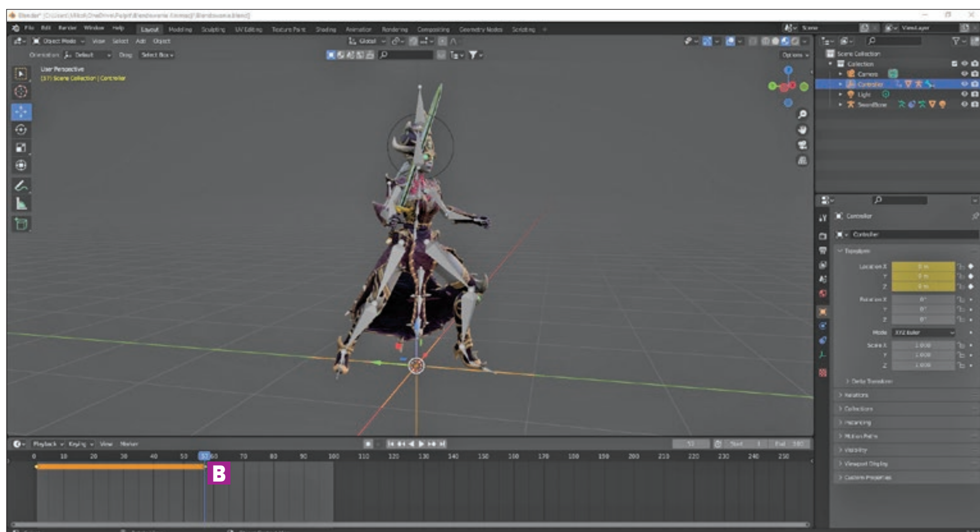
**1** Pozostaje nam dodać przesunięcie całej postaci, ponieważ ciągle porusza się w miejscu. Dodajmy nowy pusty obiekt. Nazwiemy go **Controller** **A**. Następnie ustawimy jako jego dziecko naszą postać za pomocą ctrl+P, **Object**.

**2** Musimy teraz stworzyć animację przesunięcia kontrolera, kiedy rozpoczyna się ruch. W pierwszej klatce ustawiamy już dla niego klucz dla jego obecnej lokalizacji. Przesuwamy się powoli po osi czasu **B**. W momencie, w którym rozpocznie się ruch do przodu, jeszcze raz ustawiamy klucz dla pozycji zerowej.

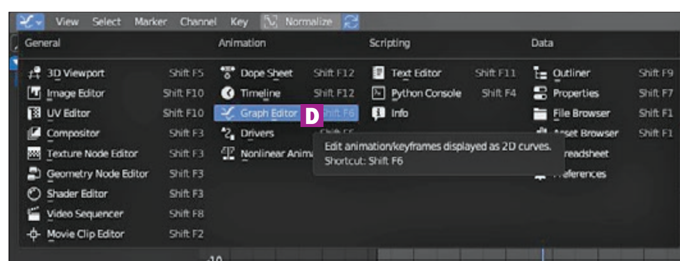
**3** Teraz ustawiamy się na ostatniej klatce **C**. Następnie przesuwamy kontroler dalej. Gdy to zrobimy, wstawiamy klucz na osi czasu.

**4** Teraz, gdy uruchomimy animację, zauważymy, że postać lekko przyspiesza na

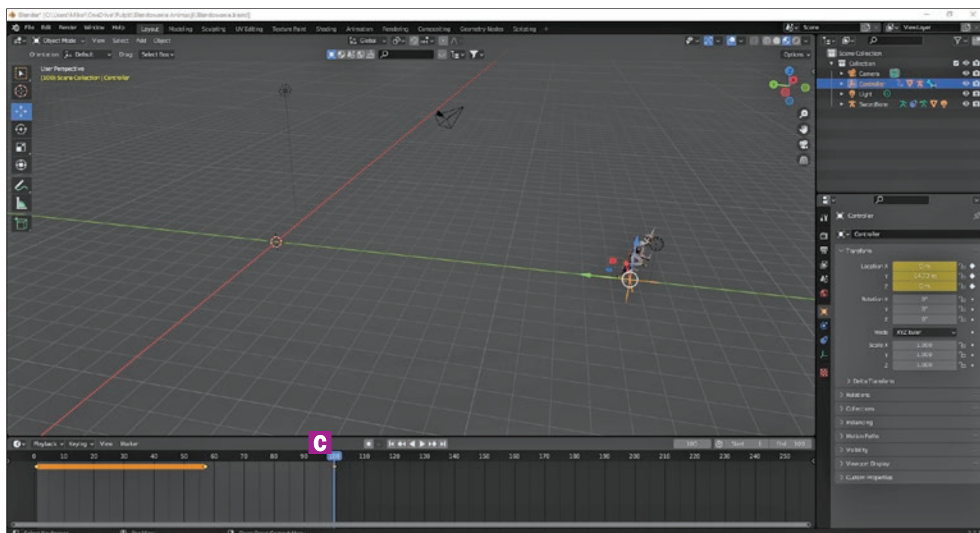


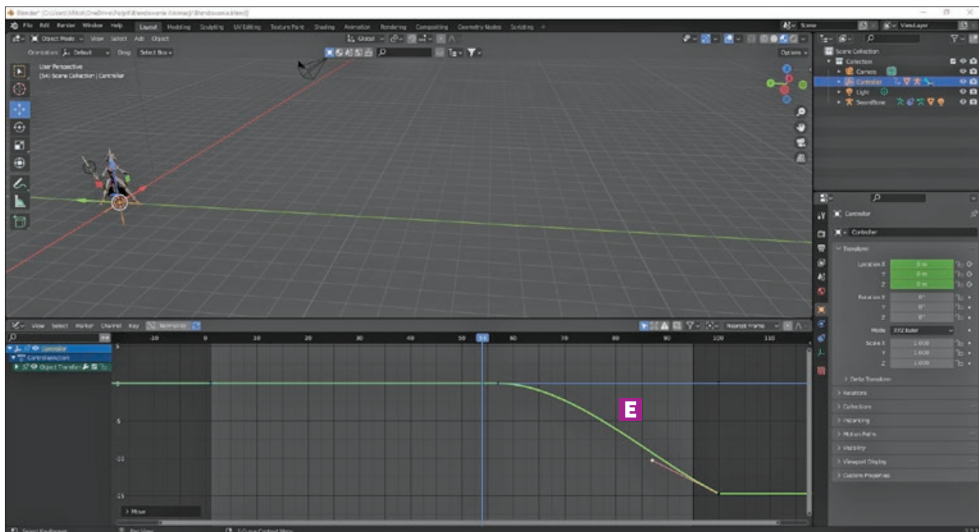



początku, a na końcu zwalnia. Warto to wyrównać. Przechodzimy do **Graph Editor** D.




**5** Powinna się pojawić krzywa E, która pokazuje zmianę prędkości





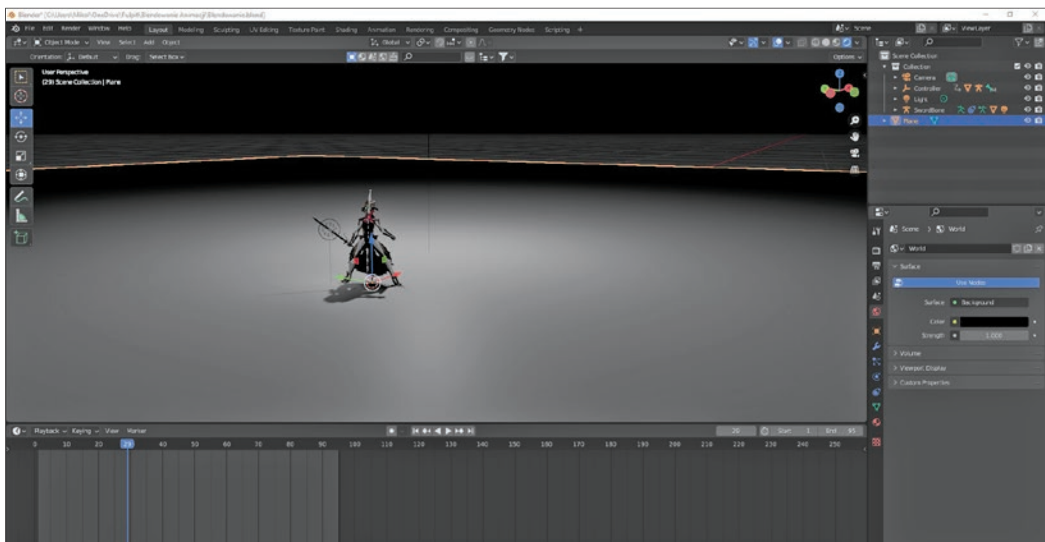

poruszania się. Jest ona delikatnie wykrzywiona. Wyprostujmy ją więc na końcu. Wystarczy, że złapiemy za końcówkę i pociągniemy w górę. Dzięki temu powinniśmy wyrównać ruch.

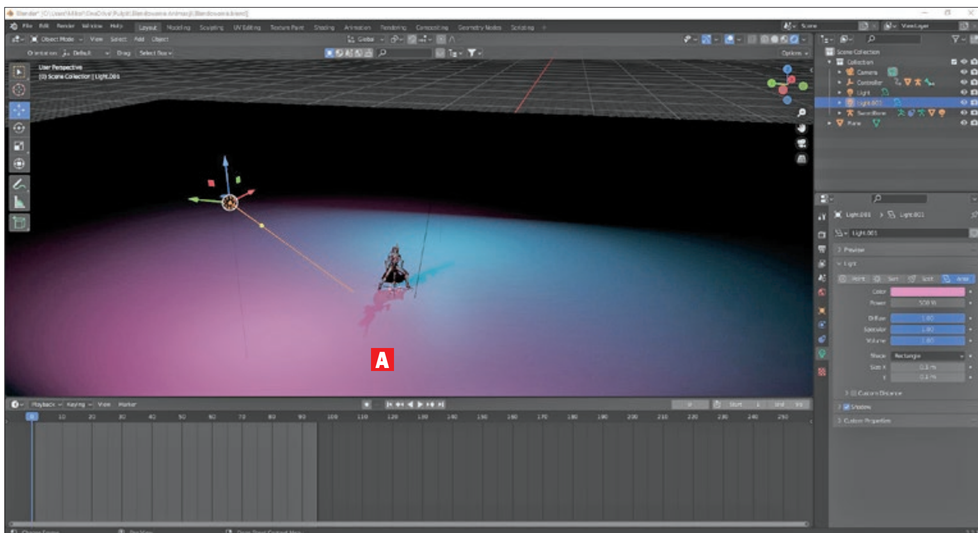
**6** Może się zdarzyć, że animacja w pewnym momencie będzie się zatrzymywać. Możemy wtedy zmniejszyć liczbę klatek (wypalona animacja może mieć trochę za dużo pustych klatek).

# Kolor i światła

**1** Pozostaje nam dodać trochę kolorów. Dodajemy **Plane** – to będzie podłoga. Ustawiamy kolor sceny na czarny.

**2** Zmieniamy teraz światło. Zamiast **Point** ustawiamy **Area**. Wybieramy kierunek i kolor. Zmniejszamy moc na **500**. Następnie

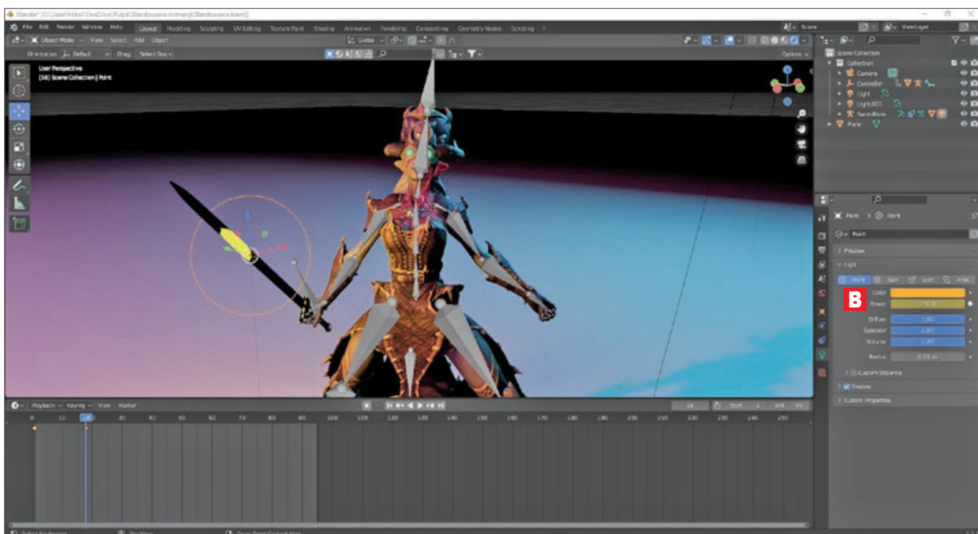




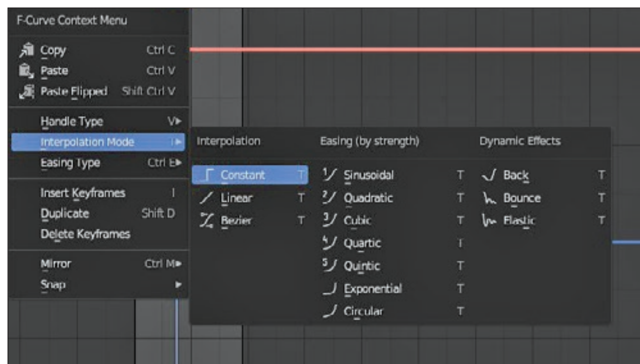

duplikujemy światło i ustawiamy z drugiej strony. Tutaj też zmieniamy kolor **A**.

**3** Pozostaje nam jeszcze światło miecza i jego blask. Zaznaczamy to światełko i w pierwszej klatce ustawiamy klucz dla **Power** na **0**. Kolor możemy ustawić na pomarańczowy. Następnie przesuwamy się na klatkę, w której będziemy chcieli włączyć moc. Wtedy ustawiamy klucz dla **Power** na **100** **B**.

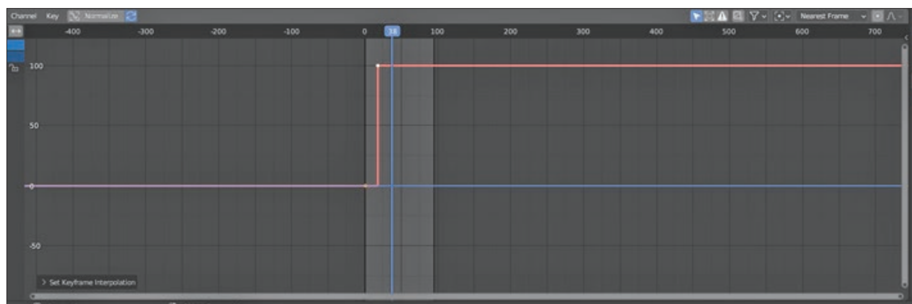
**4** Mamy jednak problem, ponieważ cały czas światło zwiększa moc. A my potrzebujemy, by całe światło włączyło się na pełną moc w jednej chwili. Musimy przejść do **Graph Editor**. Znów mamy krzywą. Usta-



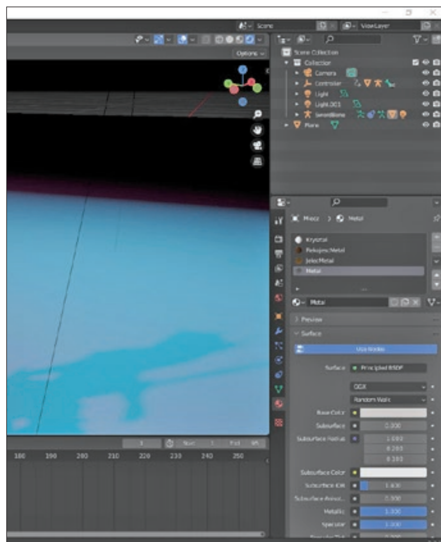
wiamy ją tak, by zmieniała się z jednej opcji w drugą natychmiast, bez powolnego przejścia. Klikamy na linię prawym przyciskiem myszy i z listy wybieramy **Interpolate Mode** i **Constant**.



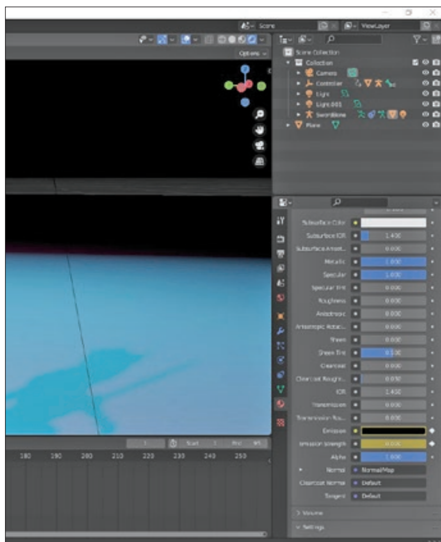

**5** Zobaczymy, że teraz nasza krzywa przybrała kanciastą formę.

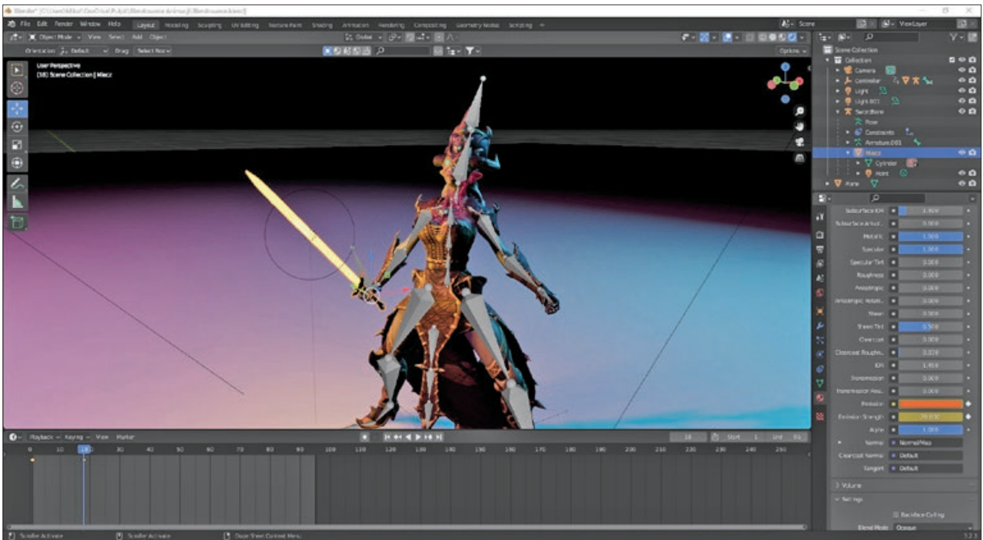


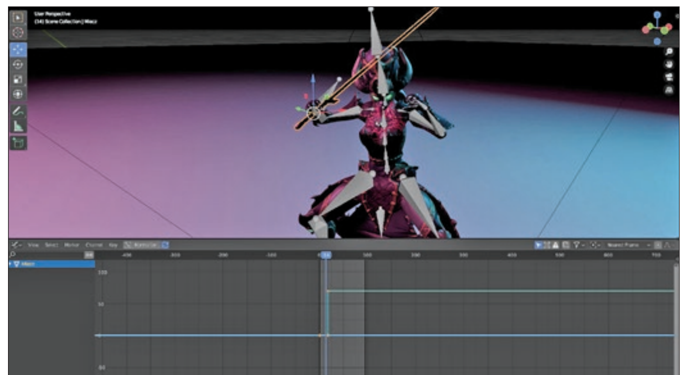

**6** To samo robimy z materiałem klingi. Zaznaczamy miecz i wchodzimy w **Material Properties**. Jest tu kilka materiałów, wybieramy ten, który nazywa się **Metal**.



**7** Zmieniamy teraz parametry **Emission** i **Emission Strength**. Na klatce pierwszej **Emission** ustawiamy na czarny, a **Strength** na **0**. Na prawo od nich znajdują się punkty, które po kliknięciu dodadzą nam klucz do klatki.



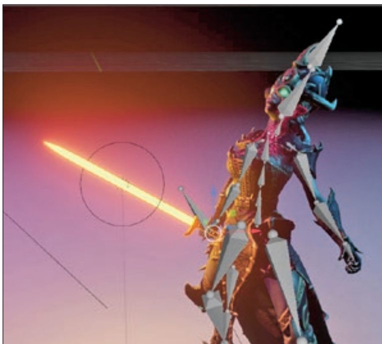




**8** Przesuwamy się na klatkę, na której włącza się światło. Wtedy zmieniamy kolor **Emission** na taki, jaki wybraliśmy w światełku, a **Strength** na **60–80**. Ustawiamy też w **Graph Editor** przejście w dokładnie ten sam sposób jak w wypadku światła.

**9** Na sam koniec przechodzimy do **Render Properties** i włączamy **Ambient Occlusion**, **Bloom** i **Screen Space Reflection**. Będzie lepszy efekt.

Jedyne, co nam teraz pozostaje, to wyrenderować naszą animację.





# 6 Dodatkowy projekt

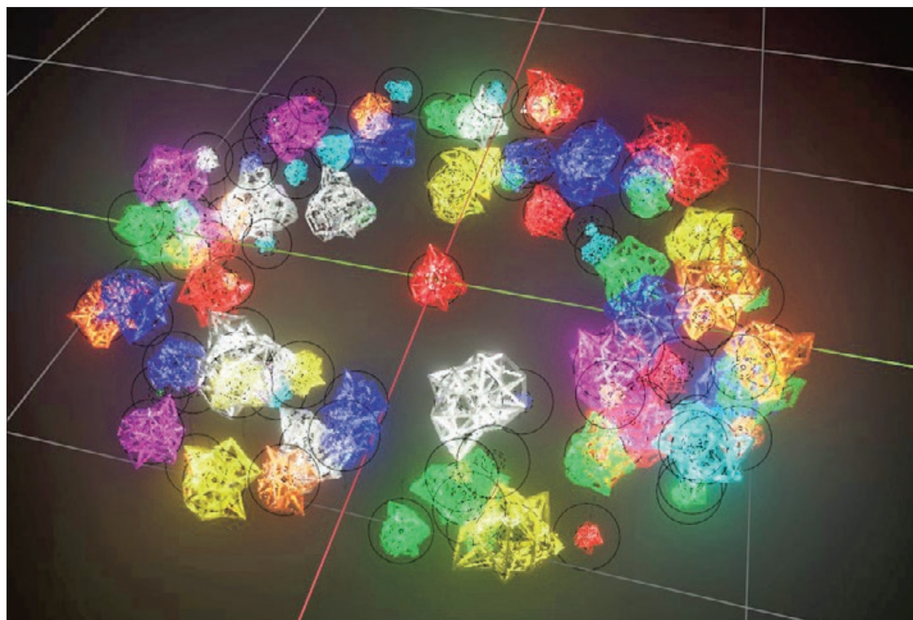



Na DVD znajdziemy plik projektu przedstawionego w tym rozdziale

**W ostatnim rozdziale – na deser – zrobimy jeszcze jeden projekt krok po kroku. Dzięki niemu poznamy ciekawą funkcję Blendera**

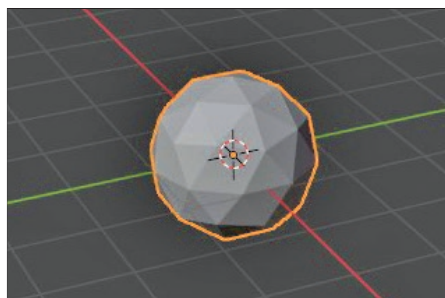
## Wizualizacja dźwięku

Blender daje ciekawe i niespodziewane możliwości. Za przykład może posłużyć dźwięk. Dźwięk w animacji kojarzy się z dubbingiem. Ale może on też wprawiać w ruch. Z fizyki pamiętamy, że dźwięk to fala, która przechodzi po ośrodku składającym się z cząstek. Te cząstki są wprawiane w ruch i dzięki temu dźwięki są słyszane z różnych odległości. Falę możemy zaprezentować w formie wykresu. A mając wykres, możemy względem niego modyfikować pewne właściwości.

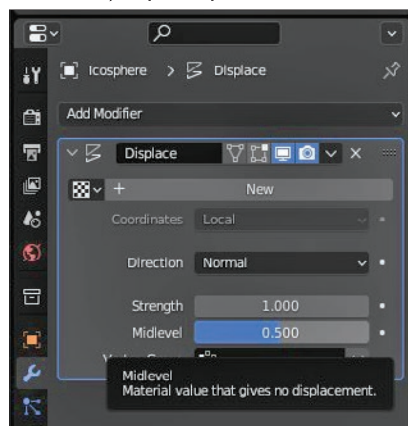


## Tworzymy nowy projekt – sfera, czyli kula

**1** Tworzymy nowy projekt w Blenderze. Usuwamy cube i wstawiamy **Ico Sphere**. Będzie ona uderzać w rytm muzyki. Oczywiście można to zrobić na dowolnym innym **meshu**.

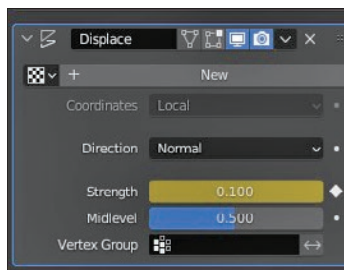


**2** Wchodzimy teraz do **Modifier Properties** i dodajemy modyfikator **Displace**.

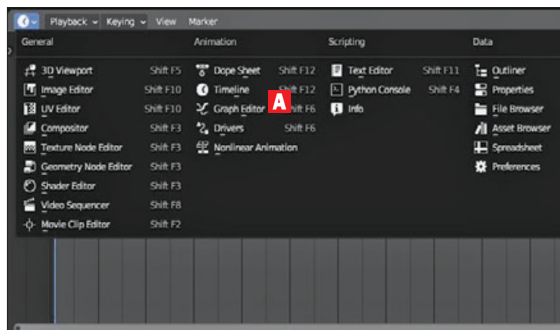



**3** Modyfikator **Displace** pozwala na przesunięcie wierzchołków na podstawie właściwości tekstury. Mamy tam modyfikator **Strength**. Jest to siła naszego przemieszczenia. Ustawiamy ją na **0.1**.

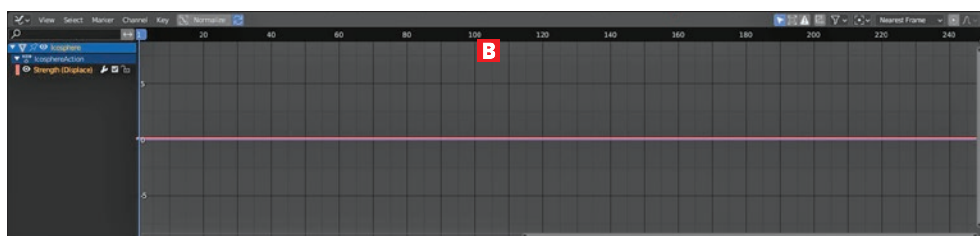
**4** Teraz przechodzimy do naszej osi czasu i ustawiamy się w pierwszej klatce. Będzie to klatka kluczowa. W modyfikatorze **Displace** klikamy na kropeczkę przy suwaku **Strength** i dodajemy nasz klucz.




**5** Przechodzimy teraz do **Graph Editor**. Rozszerzamy oś czasu. W jej lewym górnym rogu znajduje się ikona, która pozwala zmienić okno. Gdy na nią klikniemy,

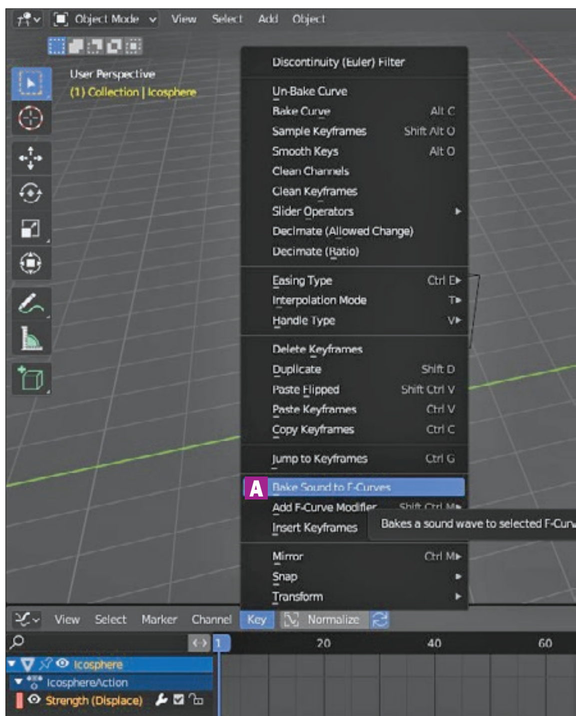



pojawi się lista. Wybieramy **Graph Editor** **A**. Otworzy się zupełnie inne okno **B**.



## Dodajemy dźwięk

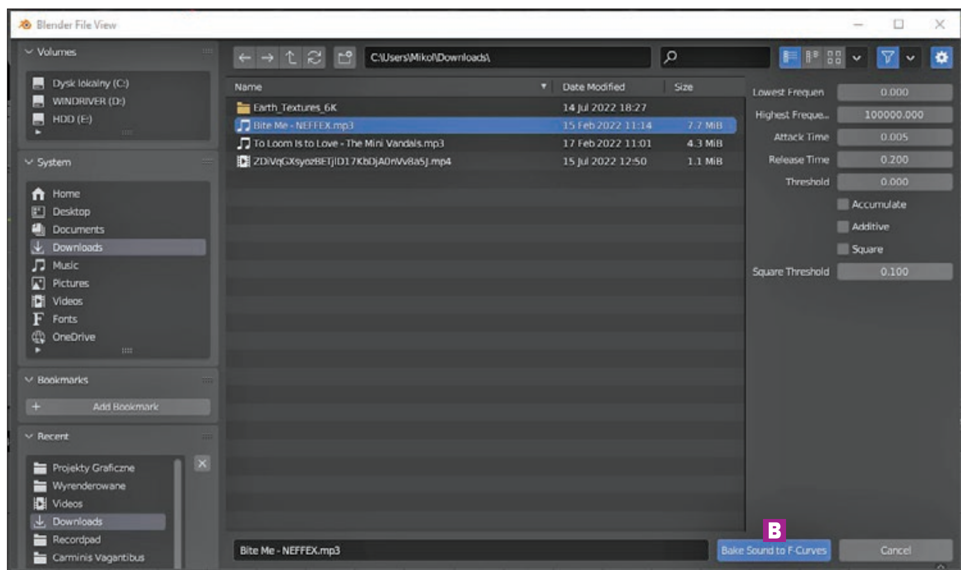
**1** Po lewej stronie widzimy listę elementów. Ponieważ mamy zaznaczoną pozycję **IcoSphere**, w której modyfikator **Displace** jest ustawiony do animowania, to dla niego pojawia się linia animacji. Wizualizuje ona, jak zmienia się wartość w trakcie animacji. I teraz ją zmodyfikujemy pod wybraną melodię. Potrzebujemy pliku dźwiękowego.
**Warto wiedzieć:** Jeżeli nie mamy żadnego pliku MP3, możemy skorzystać z konta YouTube i z Panelu twórcy. Tam jest dostępna baza darmowych dźwięków, z których można skorzystać.

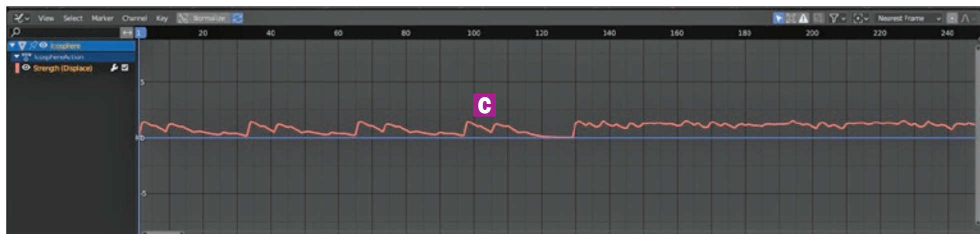



**2** Gdy mamy już pobrany dźwięk, musimy wybrać **Key**, **Bake Sound to F-Curves** **A**.

**3** Pojawi się okno wyboru, w którym musimy wybrać plik do otworzenia. Następnie klikamy na **Bake Sound to F-Curves** **B** i czekamy chwilkę.

**4** Teraz powinniśmy zobaczyć, że linia stała się pokrzywiona **C**. Jest to wizualizacja wybranego utworu.



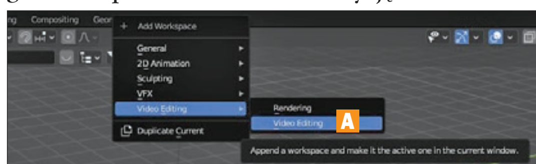


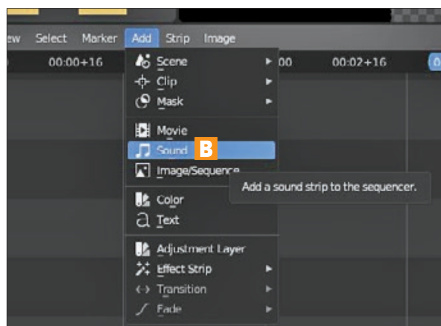

**5** Teraz wróćmy do **Timeline'u** i zobaczmy, co się stanie po uruchomieniu animacji. Nasza sfera powiększa się i pomniejsza. Można powiedzieć, że tyle wystarczy. Wybieramy wartość, dodajemy jej klatkę kluczową, a następnie wypalamy krzywą z dźwiękiem.

## Poprawiamy efekt – upiększamy kulę

Możemy podrasować nasz projekt. Naszym celem będzie zrobić serię różnokolorowych kulek, które poruszają się w przestrzeni i podskakują w takt muzyki. Zaczniemy jednak od dopracowania naszej pierwszej kuli.

**1** Najpierw, by nie było tak cicho, uruchamiamy plik dźwiękowy w Blenderze. Blender jest bardzo zaawansowanym programem i pozwala również na edycję ście-

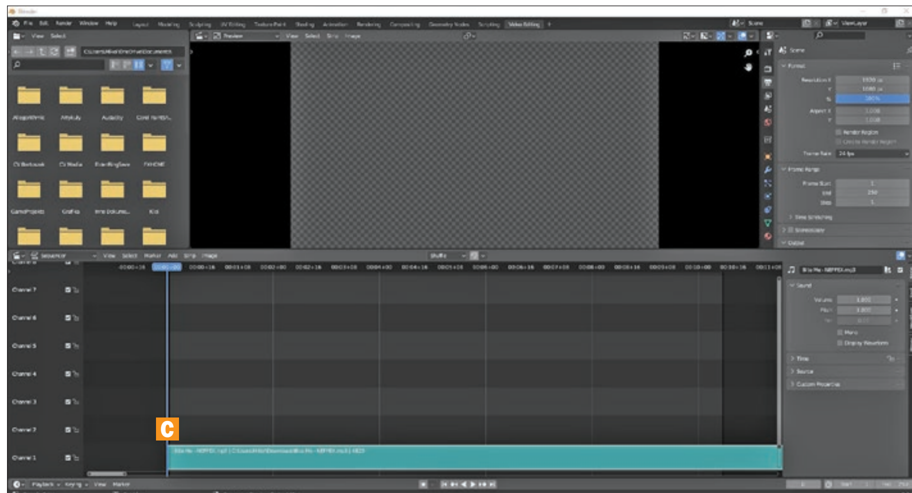



żek filmowych. Dodajmy nową zakładkę na samej górze. Klikamy na **+**, **Video Editing**, **Video Editing** **A**.

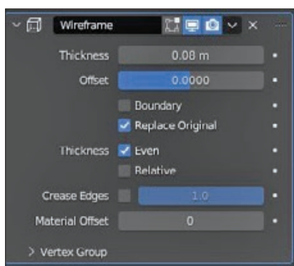
**2** Pojawi się nowe okno do miksowania różnych ścieżek. My zastosujemy tylko jeden element. Klikamy na **Add**, **Sound** **B** i znów wybieramy nasz utwór.

**3** Pojawi się obiekt na jednej ze ścieżek. Musimy go teraz ustawić na sam początek **C**, czyli na **0**, by się

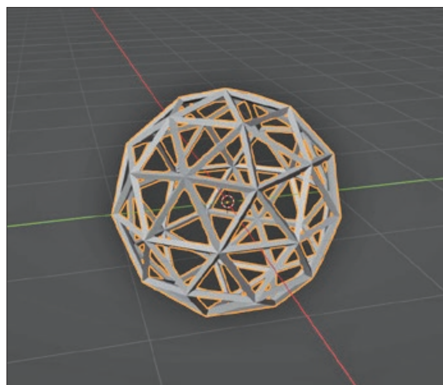


odtwarzał równo z animacją. Wystarczy, że go złapiemy i przeciągniemy do początku. Możemy teraz przejść do **Layoutu** i sprawdzić, jak to działa.

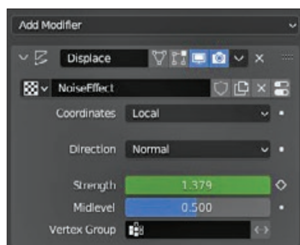
**4** Teraz czas na małe zmiany w naszej sferze. Wracamy do modyfikatorów i dodajemy modyfikator **Wireframe**.



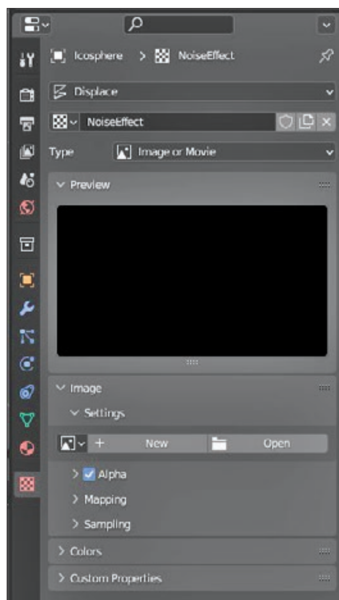

**5** Ten modyfikator pozwala na uzyskanie siatki (szkieletu) na bazie krawędzi naszego obiektu. Właściwość **Thickness** pozwoli nam na powiększenie krawędzi. Ustawiamy ją na **0.07 – 0.09**. Dobrze jest mieć włączone **Replace Original** – wtedy oryginalny obiekt dalej jest widoczny. Dzięki temu widać sam szkielet.



**6** A teraz wracamy do **Displace**. Jego działanie opiera się na teksturze, więc można by jakąś dodać. Klikamy na przycisk **New**. Okno nieco się zmieni. Pojawi się pole do wprowadzania nazwy. Zmieniamy w nim **Texture** na **NoiseEffect**.



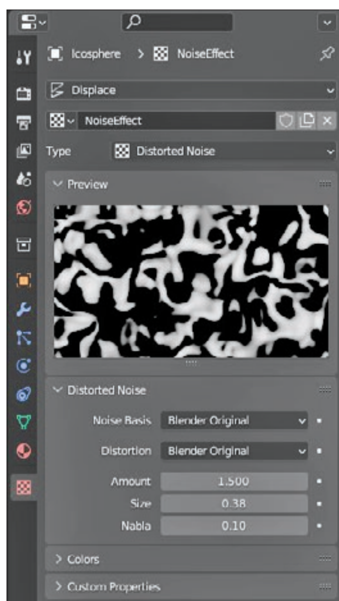

**7** Teraz powinniśmy przejść do zakładki **Texture Properties**.



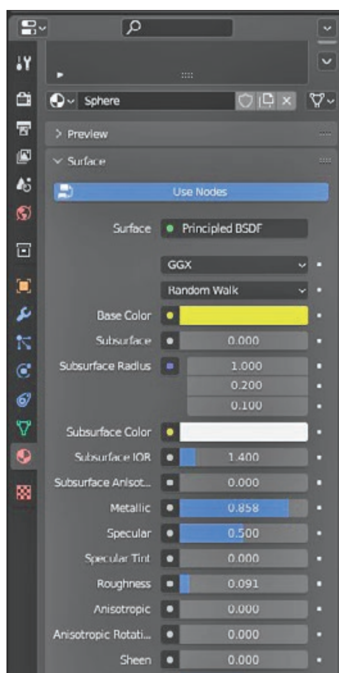

**8** Musimy tu wybrać rodzaj tekstury oraz jej właściwości. Możemy się pobawić rodzajami tekstur. Ważne, by były czarno-białe. Nasz przykład będzie działał na typie **Distorted Noise**. Właściwość **Amount** ustawiamy na **1.5**, a **Size** na **0.38**.

**9** Zobaczmy, jak teraz wygląda efekt. Z powodu dodania tekstury wierzchołki naszej sfery będą się przenosić na różne wysokości.

**10** Dodajemy teraz materiał do naszej kuli. Wybieramy taki kolor, jaki nam

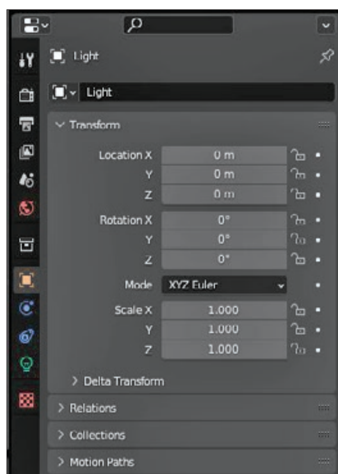



się podoba, w **Base Color**, **Metalic** ustawiamy na bliski **1**, a **Roughness** na bliskie **0**, ale nie dokładnie na 0.

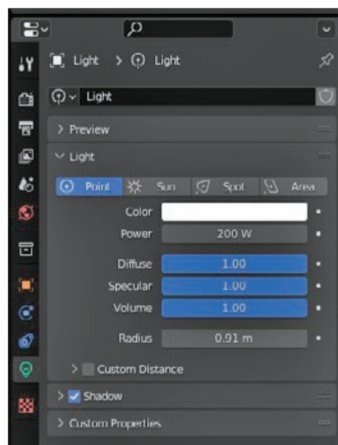



## Światło

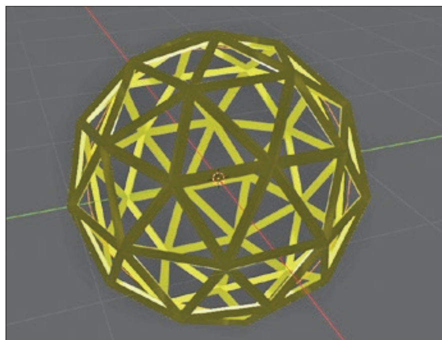
**1** Teraz zajmiemy się światłem. Będzie ono świecić od środka kuli. Klikamy więc na światło i określamy jego pozycję, by była taka jak kuli. Najłatwiej jest to zrobić, wchodząc w **Object Properties** i ustawiając dla światła te same parametry **Position** co dla kuli.



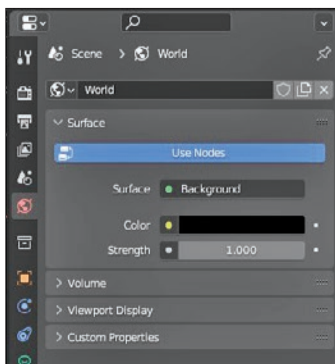

**2** Teraz przechodzimy do **Object Data Properties** dla światła. Wśród właściwości jest typ światła. Pozostawiamy światło typu **Point**, gdyż ono oświetla wszystko dookoła siebie. Jest też kolor światła (**Color**). Zostawiamy biały. **Power** to intensywność światła – powinna wystarczyć wartość **200**. Z pozosta-

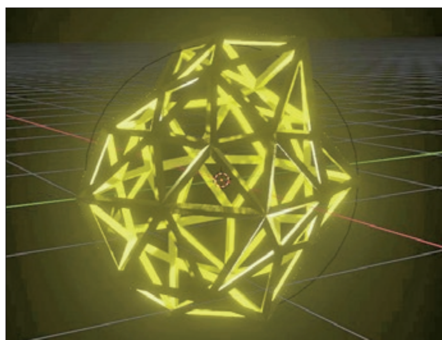


łych właściwości ważny dla nas jest **Radius**, czyli promień światła. Powiększamy go tak, by kula była wewnątrz zawsze oświetlana. By był nieco mniejszy od kuli w jej pełnym kształcie.

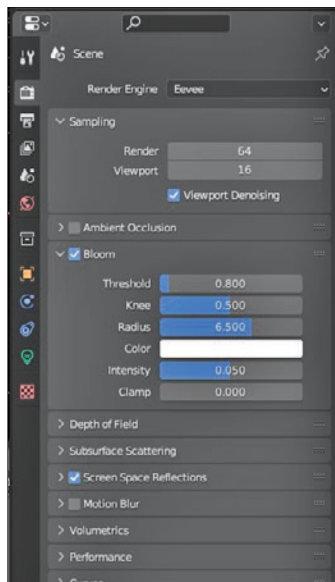


**3** By podkręcić klimat, przechodzimy do **World Properties** i tam zmieniamy właściwość **Color** na **czarny**. Dzięki temu w podglądzie renderu cała scena będzie czarna.



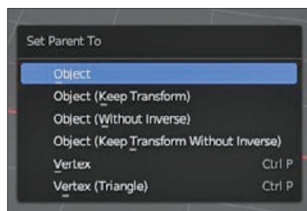
**4** Teraz w **Render Properties** włączamy **Bloom** oraz **Sceen Space Reflection**. Dzięki temu uzyskamy efekt łuny, a także odbijania światła od powierzchni modelu oraz inne odbicia.



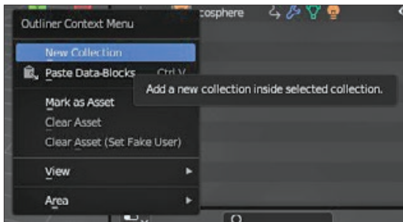

## Dodajemy więcej kul

Mamy już jedną kulę ze światłem. Aby dodać ich więcej, skorzystamy z **Geometry Node**. Jednak zanim to zrobimy, musimy przygotować dla nich scenę.

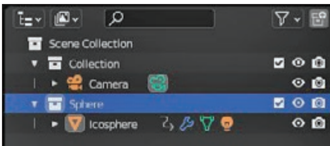
**1** Najpierw połączymy sferę i światło. Klikamy na sferę, a potem z wciśniętym klawiszem shift na światło. Gdy mamy je zaznaczone, używamy skrótu klawiaturowego ctrl+P. Z listy, która nam się pojawiła, wybieramy **Object**. I gotowe, nasza sfera i światło się połączyły i są od siebie zależne.



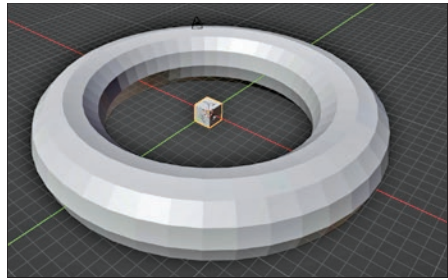
**2** Teraz przesuwamy je do nowej kolekcji. Klikamy prawym przyciskiem myszy na obszar drzewka obiektów i z listy, która się pojawia, wybieramy **New Collection**.




**3** Kolekcje w Blenderze pozwalają na zarządzanie grupą obiektów. Można kilka naraz wyłączyć, by nie przeszkadzały w trakcie pracy. Do naszej nowej kolekcji przesuwamy sferę. Wystarczy, że ją złapiemy i przeciągniemy.

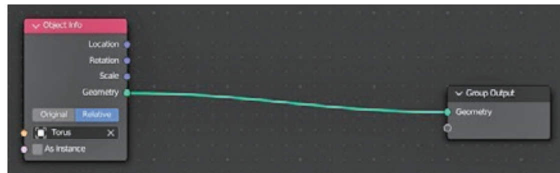



**4** Dodamy teraz obiekt, który posłuży nam jako ścieżka, po której będą się poruszać kopie naszej kuli. Możemy wybrać **Torus**. Warto go powiększyć. Nasz torus powinien się znajdować w pierwszej kolekcji, z kamerą. Dodajemy też jeszcze jeden pusty obiekt. Może to być zwykły cube. On też musi się znajdować w pierwszej kolekcji. Nazwiemy go **Geometry** i posłuży on nam jako źródło



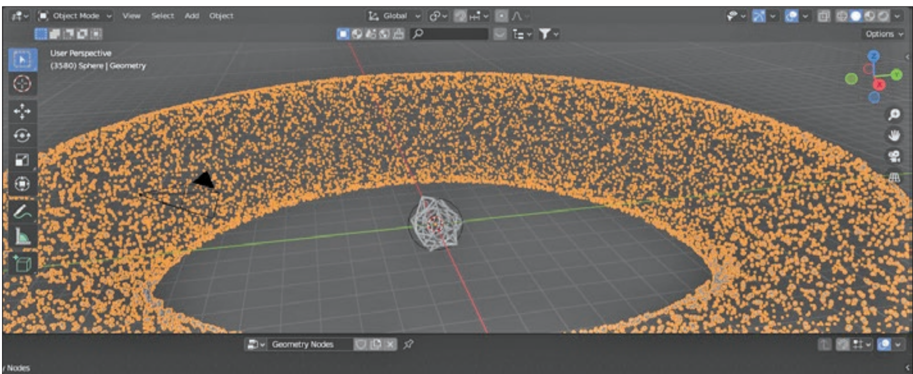
węzłów. Przechodzimy teraz do **Geometry Nodes**.

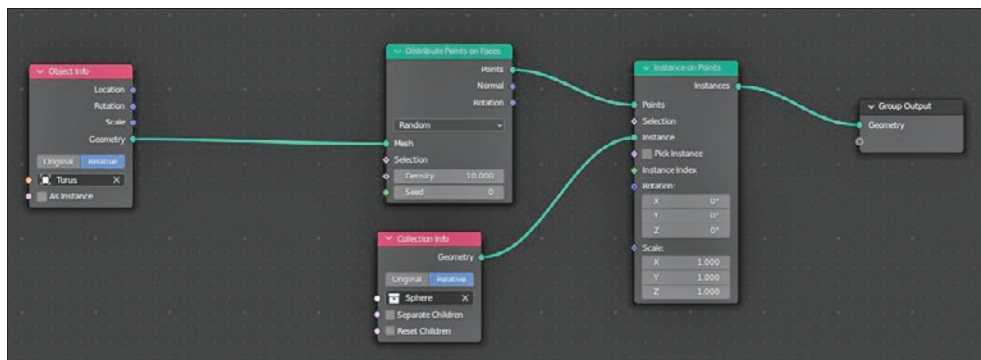
**5** Gdy już tam jesteśmy, klikamy na **New**. Usuwamy boczek **Group Input**. Na jego miejsce stawiamy węzeł **Object Info**. Pozwoli on nam na wybranie naszego torusa jako obiektu wzorcowego do pracy. Wybieramy opcję **Relative**, by był takiej wielkości, jaką




wybraliśmy wcześniej, i łączymy dwa pola **Geometry**. Dzięki **Relative** będziemy mogli zmienić wielkość torusa, zmieniając wielkość oryginału.

**6** Dodajemy teraz bloczek o nazwie **Disturbate Points on Face**. Wstawiamy go pomiędzy dwa nasze bloki. Zobaczymy, że nasz torus zamienił się w pierścień punktów.



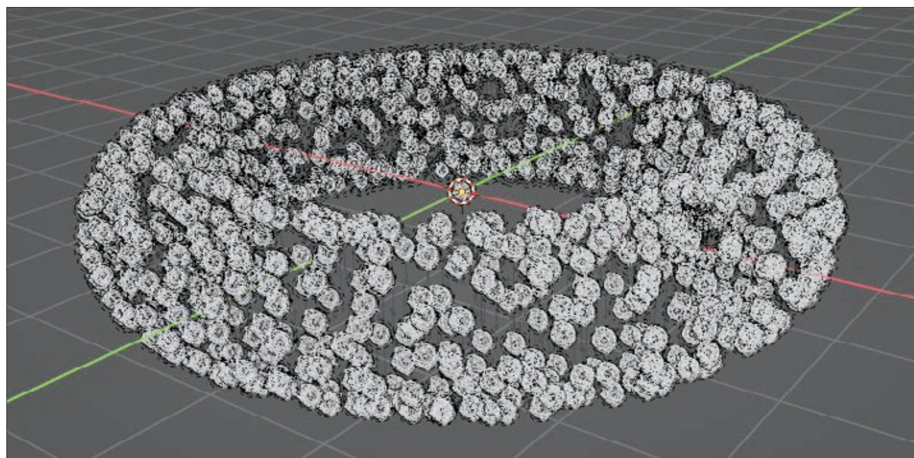


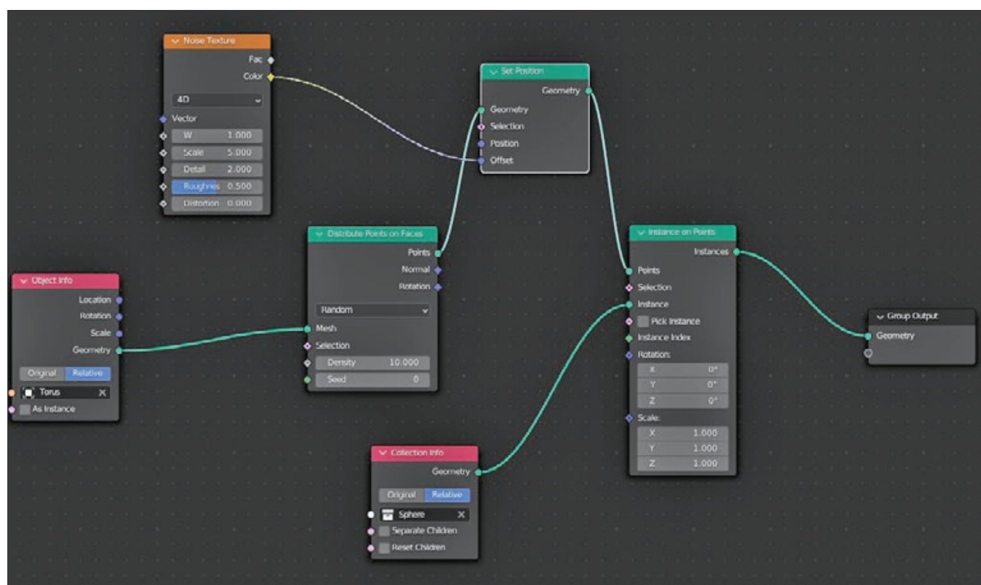

**7** Teraz będziemy chcieli w tych punktach wstawić naszą sferę. Dodajemy węzeł **Instance on Points** i wstawiamy go między **Distribute** a **Group Output**. Wszystko zniknie – musimy mu przekazać informacje, co ma zostać wyświetlone. Dodajemy bloczek **Collection Info**. W nim musimy wybrać kolekcję ze sferą. Zaznaczamy również opcję **Relative**, która będzie działać dokładnie tak samo jak w poprzednim przypadku. Gdy zmodyfikujemy wielkość oryginału, to również nasze instancje się zmienią. Łączymy teraz **Collection Info** z **Instance on Points (Geometry – Instance)**.

**8** Na naszym torusie powinny teraz się pojawić kulki. Jeżeli wydaje nam się, że jest ich za dużo, zmniejszamy torus. Nie włączamy jeszcze opcji podglądu renderu, gdyż może to jeszcze zająć wiele czasu.

## Wprawiamy kule w ruch

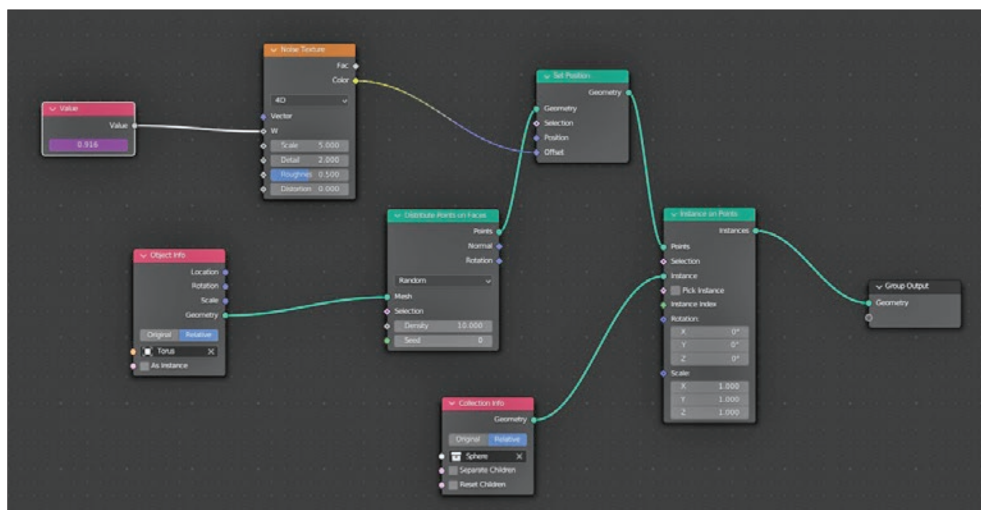
**1** Żeby sprawić, by kulki mogły się poruszać, dodamy teraz dwa bloczki. Pierwszy to będzie **Noise Texture**, a drugi – **Set Position**. W teksturze, z listy wybieramy **4D**. To pozwoli nam na przesunięcie – dzięki parametrowi **W**. Z **Noise Texture** „wyciągamy" **Color** i łączymy go z **Offset** w **Set Position**, a następnie do niego podłączamy **Points** z **Distribute**. Teraz jeszcze tylko podłączamy **Geometry** z **Set Position** do **Instance on Points** do pola **Points**. W tym momencie, gdy będziemy manipulować parametrem **W**, będziemy przemieszczać nasze kulki.





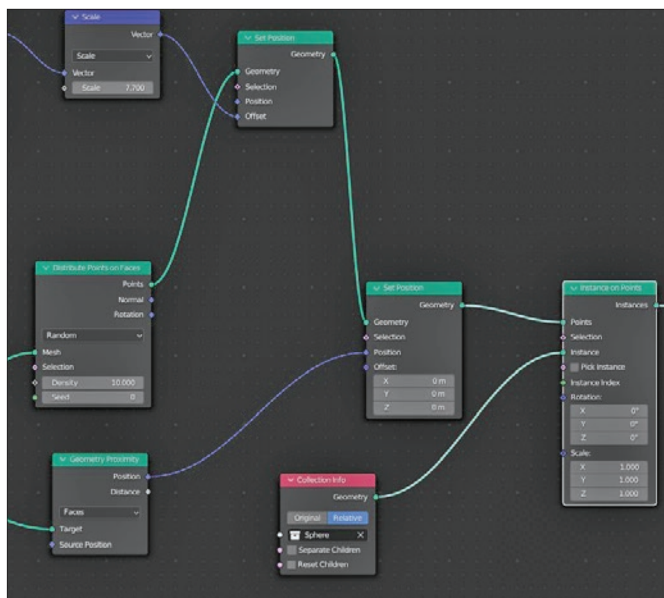

**2** By poruszanie odbywało się automatycznie, dodajemy bloczek **Value**. Podłączymy go do parametru **W** w **Noise Texture**. Teraz możemy wpisać w wartości formułkę **#frame/700**, aby wywołać animację. Przechodzimy do **Layoutu** i uruchamiamy animację, by zobaczyć, jak wygląda. Jeżeli kulki poruszają się za szybko, zamiast 700 w formułce dajemy wyższą liczbę. I analogicznie, jeśli kulki są zbyt powolne – zmniejszamy wartość.

**3** Dodamy teraz dwa bloczki typu **Vector Math**. Jeden ustawiamy na **Substract**, drugi na **Scale**. Wstawiamy je po kolei między **Noise Texture** a **Set Position**. **Substract** posłuży do przesuwania całej grupy.



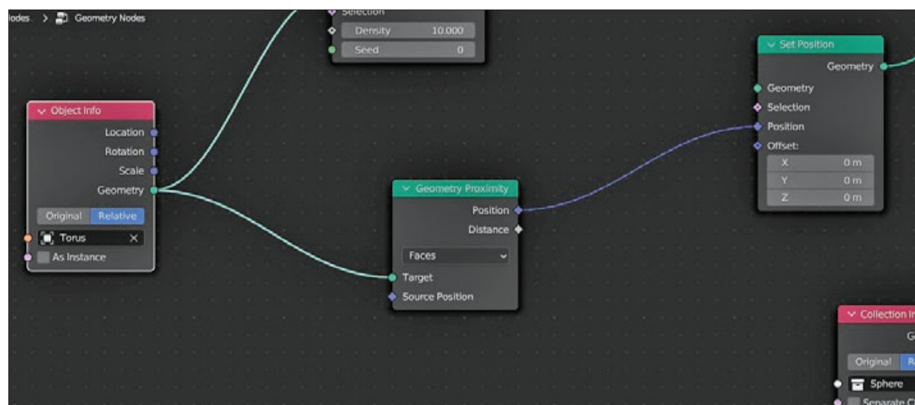


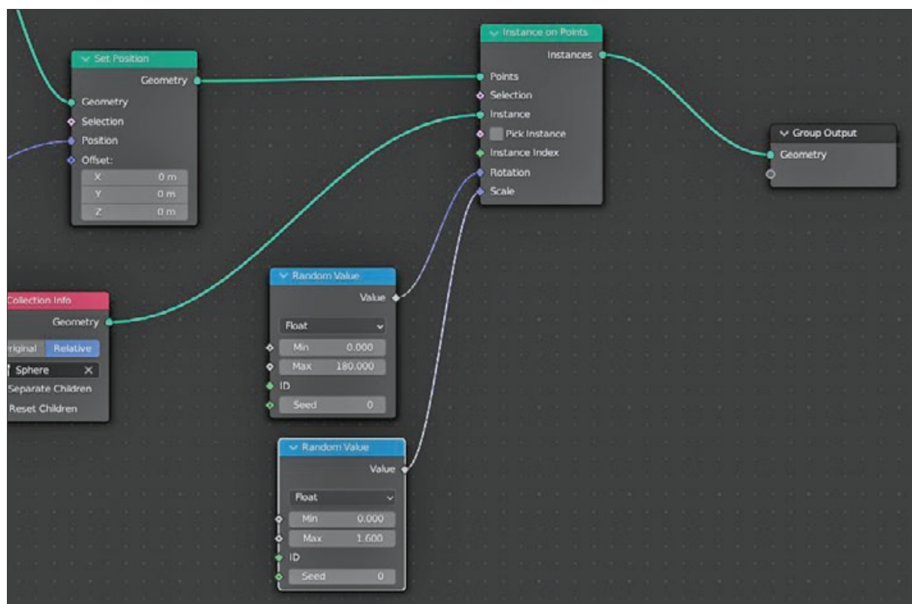
Natomiast **Scale** pozwoli na rozrzucanie kul. Im większa wartość, tym kule będą dalej odchodziły od powierzchni naszego torusa.



**4** Jeżeli zechcemy ograniczyć rozchodzenie się kul, tak by pozostały wewnątrz granic, jakie wyznacza pierścień, dodajemy dwa węzły – **Geometry Proximity** oraz **Set Position**. Do **Geometry Proximity** podłączamy **Object Info (Geometry – Target)**, a potem to ostatnie podłączamy do naszego nowego **Set Position (Position – Position)**.

**5** Następnie do nowego **Set Position** podłączamy nasze poprzednie **Set Position (Geometry – Geometry)** i to nowe łączymy z **Instance on Points (Geometry – Points)**.

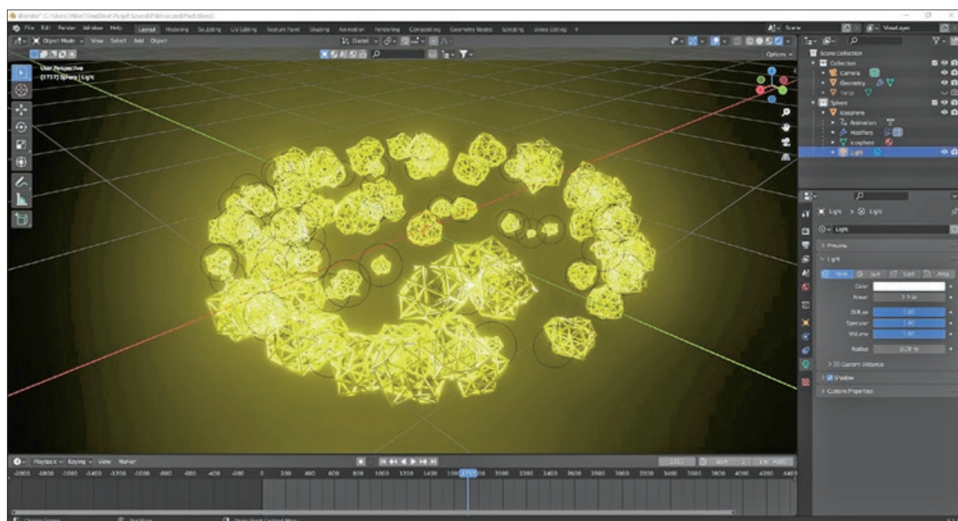


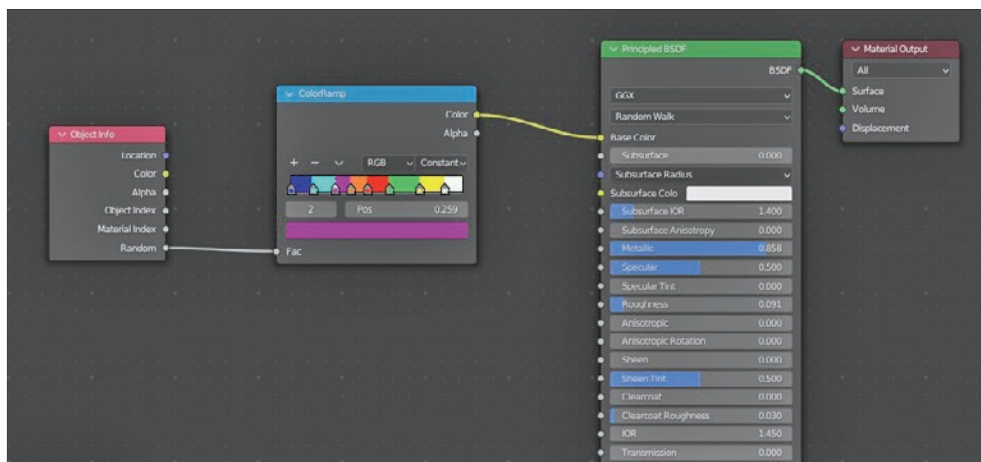



## Dla urozmaicenia – różnorodność i kolor

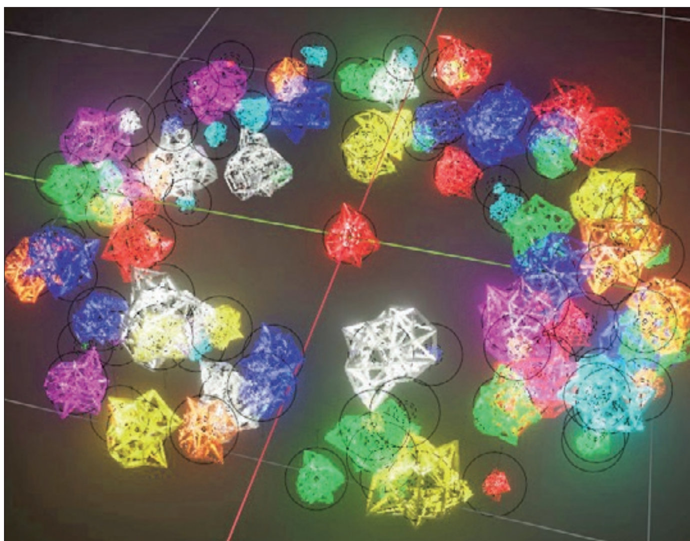
**1** Wszystko w zasadzie gotowe, ale warto jeszcze urozmaicić wygląd projektu. Dodajmy dwa bloczki **Random Value**. Pozwolą nam one na ustawienie różnych wartości dotyczących skali i obrotu kul. Podłączamy je do **Rotation** i **Scale** w **Instance on Points**. W węźle dla **Rotation** w naszym przykładzie ustawiamy wartość **Min** na **0**, **Max** na **180**, a w **Scale** – **Min** na **0** oraz **Max** na **1.6**.

**2** Wracamy do zakładki **Layout**. Jeżeli jest za dużo kulek, zmniejszamy torus. Teraz można włączyć podgląd renderu. Jeśli światło świeci za mocno, można zmniejszyć wartość **Power**.






**3** Na koniec dodajmy jeszcze kolor. Klikamy na naszą sferę i przechodzimy do zakładki **Shading**. Będziemy chcieli zrobić tak, by kulki miały różne kolory. Mamy już gotowy shader, więc możemy go trochę zmodyfikować. Dodajemy węzeł **ColorRamp**. Jego atrybut **Color** podłączamy do **Base Color** w **Principled BSDF**. Gdy to zrobimy, nasze kulki zrobią się srebrne. By uzyskać losową zmianę kolorów, musimy dodać węzeł **Object Info**. Podłączamy z niego pole **Random** do **Fac** w **ColorRamp**. Teraz nasze kulki będą miały różne kolory, od srebrnego do czarnego. W **ColorRamp** plusem dodajemy kolejne suwaki. I gdy mamy już kilka, możemy im wybrać kolor. Manipulując ustawieniem suwaków, będziemy zmieniać częstotliwość, z jaką ma się pojawić dany kolor. Jeżeli jednak za bardzo rozjeżdżają się kolory, można właściwość **Linear** zamienić na **Constant**. Wtedy będziemy mieć tylko wybrane kolory, które nie będą się mieszać.



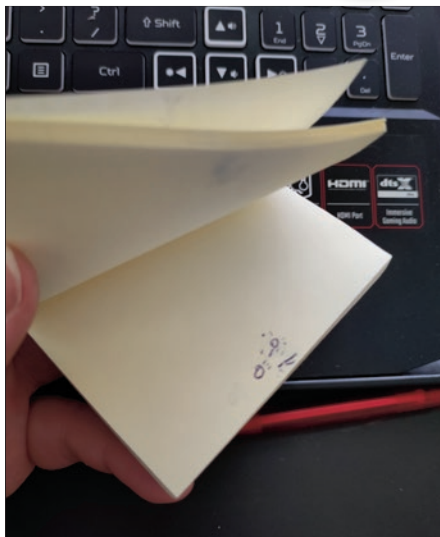
**4** To, co stworzyliśmy, wygląda fajnie i kolorowo. Jednakże wyrenderowanie tego projektu będzie trudne ze względu na liczbę klatek, którą trzeba przeliczyć, oraz liczbę źródeł światła. Jednak do odważnych świat należy :) Wiedzę tę można też łatwo zastosować w prostszych projektach, które nie będą tak obciążały komputera.

# Poznajmy FlipaClip

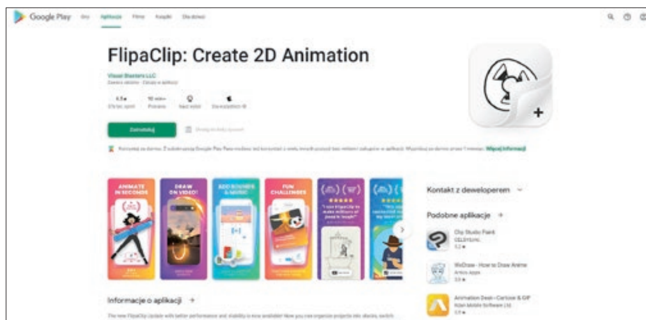


Z czasów szkolnych pamiętamy pewnie krótkie animacje robione za pomocą karteczek samoprzylepnych, na których walczyły ze sobą patyczaki albo latały statki kosmiczne? Są osoby, które tworzą w ten sposób niesamowite dzieła sztuki. Poświęcają temu dużo czasu no i przede wszystkim papieru.

Możemy takie animacje układać także za pomocą specjalnej aplikacji **FlipaClip** na urządzeniach mobilnych z Androidem bądź iOS. Jest to darmowy program do tworzenia rysowanych animacji z płatnymi dodatkami oraz rozszerzeniami. Podstawowa wersja jest jednak na tyle funkcjonalna, że na początek w zupełności wystarcza (mogą nas tylko irytować reklamy). Warto tylko mieć na uwadze,



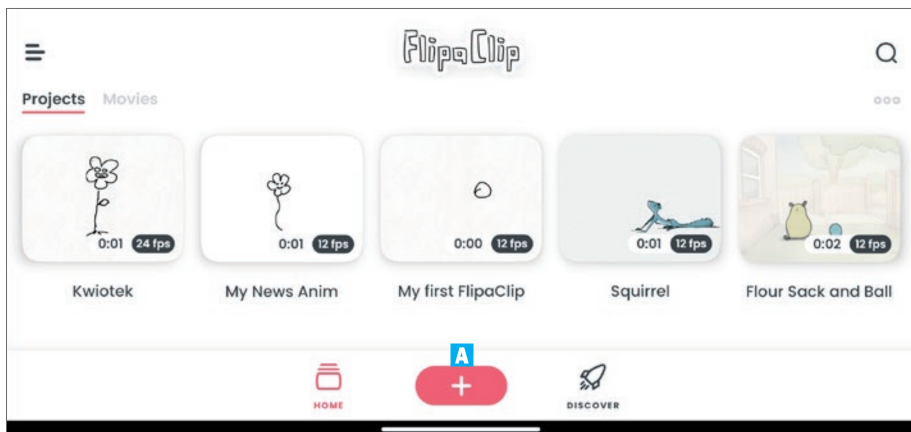
że rysunek w większości przypadków, nawet gdy nie zależy nam na profesjonalnym efekcie, wychodzi lepiej, gdy używamy rysika, nasze ręce są po prostu bardziej przyzwyczajone do takiego rysowania. Jeżeli mamy możliwość, warto zaopatrzyć się w rysik. Jeśli nie – trudno, praca palcem jest trudniejsza, ale możliwa.



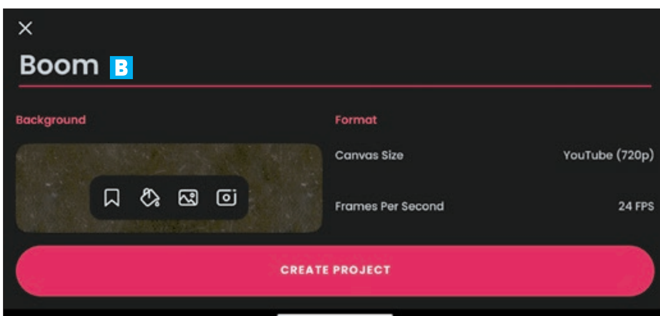

## Okno FlipaClip

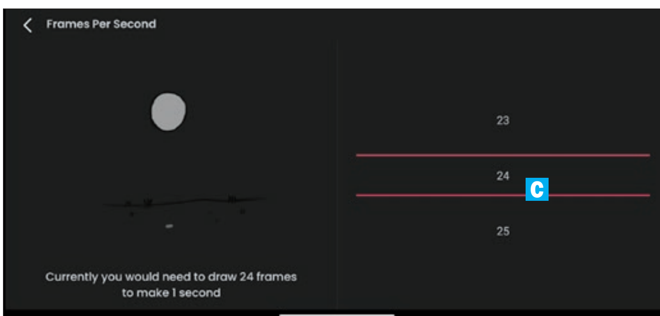
**1** Po otwarciu aplikacji pojawia się menu. Z niego mamy dostęp do wszystkich naszych projektów. Są tam też projekty poglądowe, gdzie możemy zobaczyć, co i jak jest umieszczone w projekcie. Klikając na plus **A**, tworzymy nowy projekt.

**2** Podajemy nazwę nowego projektu **B**, rozmiar projektu (standard jest ustawiony na 720p dla YouTube). Możemy też ustawić tło animacji.



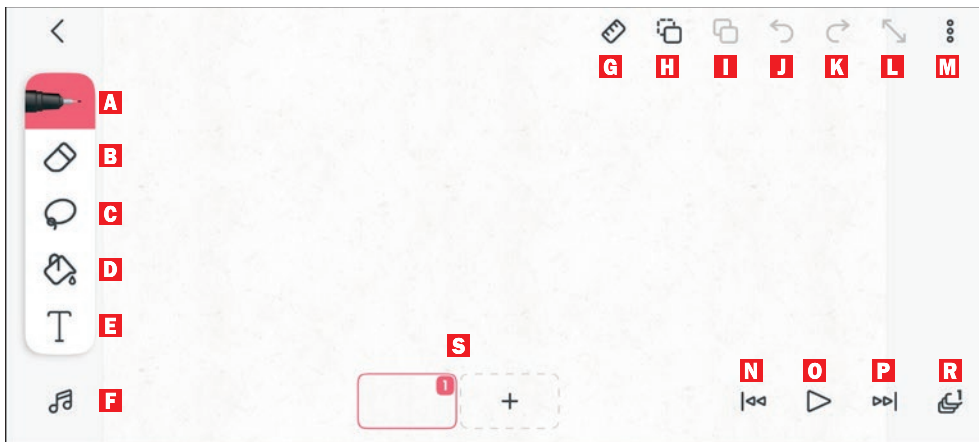

Wybieramy tu też liczbę klatek na sekundę. Pamiętajmy, że liczba klatek wpływa na płynność animacji. Możemy zwiększyć ich liczbę ze standardowych 12 do 24 **C**. Wystarczy nacisnąć i przeciągnąć suwak.



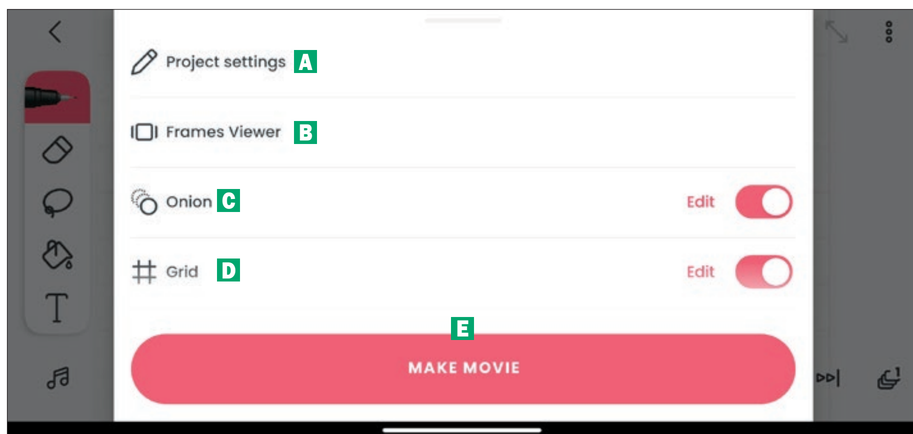

**3** Zatwierdzamy i tworzymy nasz projekt. Pojawia się nowe okno, w którym będziemy pracować. Na samym środku znajduje się część robocza, czyli ta, gdzie rysujemy. Pozostałe to (patrz następna strona):

**A** **Pisak** – możemy dla niego wybrać kolor, grubość, a nawet styl pisania (kredka, długopis, mazak).
**B** **Gumka** – wiadomo, zawsze trzeba coś wytrzeć.
**C** **Lasso** – nim możemy zaznaczać poszczególne elementy na scenie i obracać je, skalować, przesuwać czy wycinać.
**D** **Wiaderko z farbą** – gdy potrzebujemy rozlać farbę na większą powierzchnię albo pokolorować cały wybrany fragment.
**E** **Tekst** – możemy dodawać teksty, wybierać dla nich czcionkę, kolor czy wielkość.
**F** **Dźwięki** – narzędzie, które pozwala dodawać dźwięki do animacji. W wersji darmowej umożliwia jedynie dodawanie nagrywanego na bieżąco dźwięku. Paczki gotowych dźwięków trzeba dokupić, a swoje już gotowe audio można dodawać w wersji premium.
**G** **Linijka** – pozwala na rysowanie prostych linii. Może bardzo pomóc przy tworzeniu brył geometrycznych.

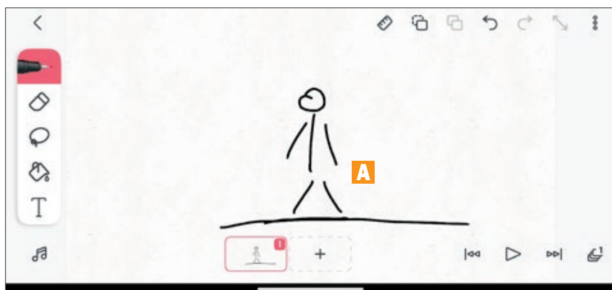


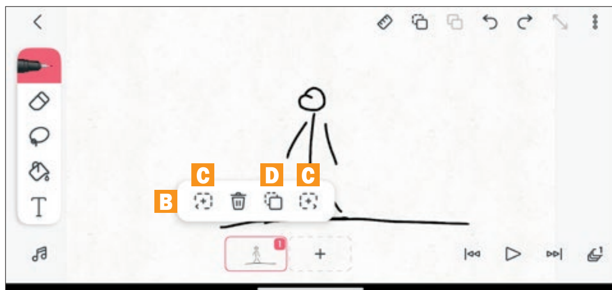

**H Kopiowanie**.
**I Wklejenie**.
**J Cofanie**.
**K Przywracanie**.
**L Przywrócenie pełnego widoku po użyciu przybliżenia**.
**M Właściwości edytora i projektu**.
**N Powrót do początku animacji**.
**O Odtworzenie**.
**P Przejście na koniec animacji**.
**R Okno warstw** – tu możemy dodawać kolejne warstwy projektu. Przydatna rzecz. Na jednej warstwie można tworzyć główny szkic, a na drugiej – dodawać kolory i efekty.
**S Linia czasu** – tu dodajemy kolejne klatki animacji i przeklikujemy się między nimi.

**4** Gdy klikniemy na właściwości edytora, otworzy się okno, gdzie będziemy mogli edytować kilka elementów.
**A Project Settings** otworzy nam pierwsze okienko. Możemy dodać tło lub zmienić liczbę klatek na sekundę.
**B Frames Viewer** włącza tablicę, na której mamy podgląd wszystkich naszych klatek.
**C Onion** to właściwość, która pozwala na podgląd poprzedniej klatki. Bardzo przydatna. Gdy narysujemy pierwszą klatkę i będziemy chcieli zrobić następną, nie będziemy musieli przeskakiwać między nimi, by ustalić, gdzie była głowa postaci itp. Będzie pokazywała w lekko wyszarzony sposób poprzednią klatkę. To znacznie polepszy komfort pracy.



**D Grid** – pozwala na włączenie siatki. Przydatne, bo łatwiej wtedy wyznaczyć niektóre granice. **Add Image** i **Add Video** – gdy chcemy do naszej animacji wprowadzić jakiś gotowy plik (po przewinięciu menu pod opcją **Grid**). **E Make Movie** – gdy mamy już gotowe klatki, pozwoli nam na wyrenderowanie ich do filmu.










## Animowany patyczak

**1** Narysujmy teraz patyczaka. Stworzymy krótką animację rozrywania go przez kulę armatnią. Na początek głowa, tułów, nogi i ręce **A**. Przyda się też ziemia.

**2** Gdy mamy go narysowanego, czas na drugą klatkę. By jednak nie tracić czasu na ponowne rysowanie, po prostu skopiujemy tę już gotową. Naciskamy i przytrzymujemy na osi czasu aktywną klatkę. Pojawi się wtedy małe menu **B**.

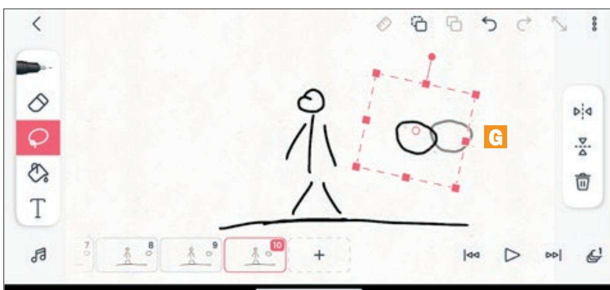
**3** Pośrodku mamy kosz do usuwania danej klatki. Na obu końcach widać przyciski z przerywanymi kwadracikami z plusami i strzałką w jedną lub drugą stronę **C**. Te przyciski dodają pustą klatkę z przodu lub z tyłu. Pozostaje ostatni przycisk **D**. On kopiuje całą naszą klatkę. Gdy go naciśniemy, małe menu się rozszerzy.

**4** Pojawiają się dwa nowe przyciski ze strzałkami bez plusów **E**. Służą one do wklejania skopiowanej klatki z prawej lub lewej strony klatki aktywnej. Użyjemy tego przycisku, który kopiuje na prawo. Stwórzmy tak trzy klatki. Na trzeciej pojawi się kula armatnia.

**5** Na początek pojawi się tylko skrawek kuli **F**. W następnej klatce kolejny kawałek aż w końcu cała. Na razie trzeba będzie usuwać poprzednio narysowany fragment kuli i dorysowywać nowe fragmenty.

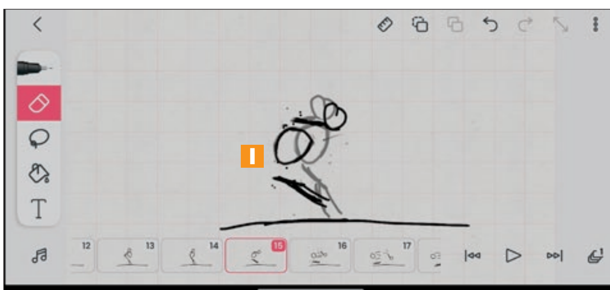
**6** Dopiero gdy pojawi się cała kula, będziemy mogli za pomocą lassa w kolejnych klatkach przesuwać ją całą G.

**7** Naszego patyczaka nie ruszamy, dopóki nie trafi w niego kula. Dopiero wtedy zaczniemy go modyfikować. Zwróćmy uwagę, że kula, mając swoją prędkość i masę, wpłynie na postawę naszej postaci, wyginając ją, a potem przecinając na pół H. Gdy postać zostanie rozerwana, poszczególne jej części mogą polecieć w różne strony. Kilka pierwszych klatek będzie wyginało naszą postać. Wtedy naszemu patyczakowi gumkujemy tułów I i rysujemy na nowo. Głowę czy nogi możemy przesunąć lassem J.













**Warto wiedzieć:** Pamiętajmy, że obraz możemy przybliżyć. Wykonujemy wtedy odpowiedni gest na ekranie. Przykładając dwa palce i rozsuwając je po ekranie, będziemy przybliżać lub oddalać widok. Przesuwając oba, będziemy się przesuwać po ekranie. A obracając je, będziemy obracać naszą kartkę.

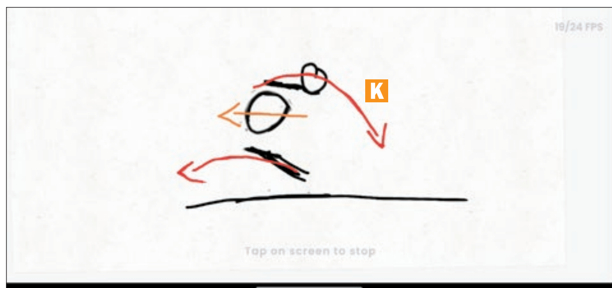
**8** Gdy już kula uderzy i rozerwie naszą postać, jej górna część poleci w inną stronę niż dolna. Rozerwane części będą dalej miały energię kinetyczną kuli, która z czasem będzie się wytracać. Dodatkowo jest też grawitacja, która ściąga wszystko w dół, więc będziemy musieli narysować w kolejnych klatkach ruch części postaci po łuku K.

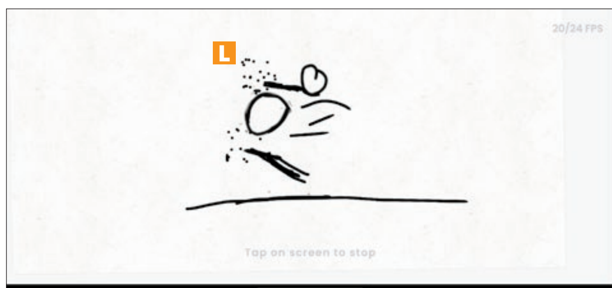
**9** W końcu wszystkie części opadną na ziemię. A nasza kula zniknie za krawędzią kartki. Możemy teraz dodać elementy wizualne. Do kuli dodamy linie pokazujące

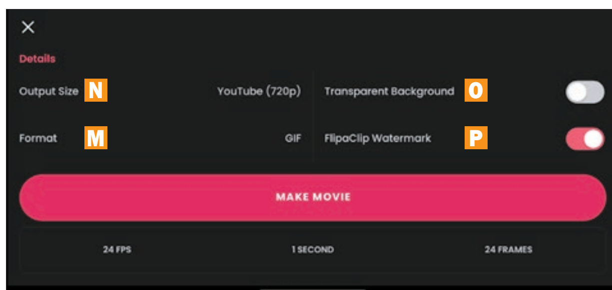
ruch powietrza, a naszemu patyczakowi przy rozrywanych częściach dorysujemy kropki **L**, by wzmocnić efekt. Można też dodać kolorów czy przygotować tło.

**10** Teraz przyszedł czas na renderowanie animacji. Naciskamy trzy kropki ustawień i wybieramy **Make Movie**. Otworzy się okno, w którym ustawimy właściwości końcowe, czyli **Format** **M** pliku (do wyboru mamy **MP4**, **GIF** i **PNG**, z tym że ten ostatni to wyrenderowanie każdej klatki osobno). **Output Size** **N** to wielkość, na przykład **Full HD**. **Transparent Background** **O** usuwa tło, czyli jeżeli nic na nim nie narysowaliśmy, to będzie ono przezroczyste. Dzięki temu będziemy mogli dodać film do innego projektu bez konieczności usuwania tła. Jeżeli nie włączymy tej opcji, to tło będzie białe. **FlipaClip Watermark** **P** możemy wyłączyć tylko wtedy, gdy mamy wykupioną wersję premium.

**11** Naciskamy **Make Movie** i czekamy. Gdy wszystko będzie gotowe, możemy przesłać animację na **TikToka** czy **YouTube**.






# JAK SKORZYSTAĆ Z E-WYDANIA KSIĄŻKI

W KŚ+ znajdziemy **e-wydanie tej Biblioteczki, obraz ISO dołączonej do niej płyty z najlepszymi narzędziami do tworzenia animacji i plik PDF książki do pobrania.**

**1** Otwieramy stronę **ksplus.pl**. Logujemy się **A** (używamy konta z serwisu **Komputerswiat.pl**). Jeżeli nie mamy konta, klikamy na **B**, by się zarejestrować.

B Załóż konto Logowanie A
Zarejestruj kod

**2** Po zalogowaniu się możemy zarejestrować kod nadrukowany na płycie dołączonej do książki. Wystarczy kliknąć na **C** i przepisać kod.

Moje konto
C Zarejestruj kod

**3** Uzyskamy w ten sposób dostęp do e-wydania **D** i do bonusowego obrazu płyty **E**. Do serwisu KŚ+ możemy logować się z dowolnego urządzenia z dostępem do internetu.

CZYTAJ E-WYDANIE D
PROGRAMY E
BONUSY

**UWAGA! W KŚ+ ZA DARMO E-WYDANIE KSIĄŻKI ORAZ PLIK ISO PŁYTY**

# POLECAMY INNE NASZE KSIĄŻKI




## WSKAZÓWKI DO OFFICE

Triki, jak w pełni wykorzystać możliwości nowoczesnego pakietu Microsoftu: Teams, Excel, OneDrive, PowerPoint, Word, Outlook. Przydadzą się w pracy i szkole! Na DVD i w KŚ+: wersje próbne Office i darmowe alternatywy.




## JAK TWORZYĆ STRONY WWW

Wprowadzenie do tworzenia stron WWW: od najprostszych sposobów na blog, przez ABC WordPressa, po podstawy programowania w JavaScripcie i pisania aplikacji webowych. Na DVD i w KŚ+: narzędzia dla programistów.

**Nasze książki w wersji drukowanej kupisz na literia.pl**
**Książki są również dostępne w formie e-wydań na ksplus.pl**



**Mikołaj Bartoszek**
**autor książki**
informatyk, nauczyciel w Gigantach Programowania i w III LO w Zabrzu




## SZYBKIE WEJŚCIE DO ŚWIATA ANIMACJI

Każdy, kto gra w gry komputerowe, docenia znaczenie animacji. Ruch mieczem w Elden Ring, kopnięcie piłki w FIFA czy tańce w Fortnite, które cieszą nasze oczy, są efektem pracy twórców animacji.

Miłośnicy kina także kochają animacje, bo wzruszające całe rodziny kreskówki czy zapierające dech w piersiach efekty specjalne w filmach fabularnych to również dzieła animatorów.

Przy obecnych możliwościach domowych komputerów świat animacji jest otwarty dla każdego. Wystarczy darmowy program, jak opisywany Blender, trochę ćwiczeń i odpowiedniej wiedzy, by zrobić pierwsze kroki jako animator 3D.

W tym zadaniu pomaga właśnie ta książka: praktyczne porady opisane krok po kroku, pliki szkoleniowe dołączone do wydania i gotowe do zainstalowania oprogramowanie sprawiają, że swoich sił w animacji komputerowej może spróbować każdy.
Zapraszam, to niesamowita przygoda!

**CENA 18,90 ZŁ**
W TYM 5% VAT

Płyta DVD jest dodatkiem do książki

ISBN 978-83-8250-199-5 INDEKS 321 958




Nr 5/2022 (121)



KOMPUTER ŚWIAT BIBLIOTECZKA